Azad M. Madni · Barry Boehm
Roger G. Ghanem · Daniel Erwin
Marilee J. Wheaton *Editors*

Disciplinary Convergence in Systems Engineering Research

Springer

Disciplinary Convergence in Systems Engineering Research

Azad M. Madni • Barry Boehm
Roger G. Ghanem • Daniel Erwin
Marilee J. Wheaton
Editors

Disciplinary Convergence in Systems Engineering Research

Springer

Editors
Azad M. Madni
Professor, Astronautics, Executive
Director, Systems Architecting
and Engineering Program
University of Southern California
Los Angeles, CA, USA

Barry Boehm
Professor, Computer Science,
Director, Center for Systems and Software
Engineering
University of Southern California
Los Angeles, CA, USA

Roger G. Ghanem
Professor, Aerospace
& Mechanical Engineering
University of Southern California
Los Angeles, CA, USA

Daniel Erwin
Professor, Astronautics
University of Southern California
Los Angeles, CA, USA

Marilee J. Wheaton
Systems Engineering Fellow
The Aerospace Corporation
El Segundo, CA, USA

ISBN 978-3-319-62216-3 ISBN 978-3-319-62217-0 (eBook)
https://doi.org/10.1007/978-3-319-62217-0

Library of Congress Control Number: 2017955102

Printed on acid-free paper

This Springer imprint is published by Springer Nature
The registered company is Springer International Publishing AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

Preface

Innovation growth in the twenty-first century is continuing to fuel the way we live, work, learn, and entertain. This era augurs well for societal well-being so long as we make the understanding and management of complexity a top priority. Specifically, the impact of innovation needs to be studied with regard to unintended consequences. The latter is a challenge for complex systems engineering and a fertile ground for conducting systems engineering research.

According to the World Economic Forum, we are in the early stages of the Fourth Industrial Revolution. Coming on the heels of the Third Industrial Revolution, which produced dramatic advances in electronics, computers, communications, and information technology, the Fourth Industrial Revolution is going to be an era of convergence. Increasingly, we are beginning to see the convergence of engineering with behavioral and social sciences, entertainment and cinematic arts, biology, and the physical sciences.

At the same time, systems in the twenty-first century are becoming increasingly hyper-connected and more complex. Recognizing that traditional systems engineering methods, processes, and tools no longer suffice, the research community supported by government, academia, and industry has begun working together to transform systems engineering. Central to this transformation is exploiting innovation and capitalizing on convergence to develop new approaches, methods, and tools. The emphasis is on reaching beyond traditional engineering to address problems that appear intractable when viewed solely through an engineering lens. Today disciplinary convergence is beginning to play a key role in this transformation.

> "....The central idea of disciplinary convergence is that of bringing concepts, thinking, and approaches from different disciplines in conjunction with technologies to solve problems that appear intractable when viewed through the lens of a single discipline." (Madni, A.M. Transdisciplinary Systems Engineering: Exploiting Convergence in a Hyper-Connected World," Springer, 2017)

This vision inspired the central theme of 2017 Conference on Systems Engineering Research (CSER): *Disciplinary Convergence: Implications for Systems Engineering Research*. This volume is a collection of peer-reviewed research papers from university, government, and industry researchers who participated in 2017 CSER. To help the reader conveniently navigate this volume, the papers are organized into ten sections. Each section represents a key research area in systems engineering research today.

It is our hope that this volume will get you interested in systems engineering research that exploits disciplinary convergence and pursues cross-disciplinary approaches to solve complex scientific and societal problems.

Los Angeles, CA, USA

Azad M. Madni
Barry Boehm

Contents

Part I
Engineered Resilience and Affordability

Chapter 1
Engineering Resilience for Complex Systems

Colin Small, Gregory Parnell, Ed Pohl, Simon Goerger, Bobby Cottam, Eric Specking, and Zephan Wade

Abstract In recent years there has been an increased need for resilience in complex military and civilian systems due to evolving adversarial and environmental threats. Engineered Resilient Systems (ERS) is a Department of Defense (DoD) program focusing on the effective and efficient design and development of complex engineered systems. These complex systems need to be resilient to threats throughout their life cycle. However, most current engineering resilience literature focuses on systems with a single function and a single measure. Today's systems are becoming more complex, with multiple functions and measures involving critical trade-offs during early life cycle stages. This paper develops criteria for a framework to incorporate resilience into DoD analysis of alternatives (AoA). Using the criteria, this paper creates a framework for defining and evaluating complex engineered systems that consider many missions, scenarios, uncertainties, functions, and measures. Lastly, using the criteria and the framework, the current literature is shown to have gaps for incorporating resilience into DoD AoAs.

Keywords Resilience • Engineering Resilient Systems • Resilience cycle • Systems engineering • DoD • Analysis of alternatives

1.1 Introduction

In recent years there has been an increased need for resilience in complex military and civilian systems due to evolving adversarial and environmental threats. As systems become increasingly interconnected and technology advances more quickly, it becomes harder for systems to resist threats. Often systems are used in unplanned missions or new scenarios with different threats. Therefore, systems

C. Small (✉) • G. Parnell • E. Pohl • B. Cottam • E. Specking • Z. Wade
University of Arkansas, Fayetteville, AR, USA
e-mail: cxs050@uark.edu; gparnell@uark.edu; epohl@uark.edu; bjcottam@uark.edu; especki@uark.edu; zwwade@uark.edu

S. Goerger
U.S. Army Engineer Research Development Center (ERDC), Vicksburg, MS, USA
e-mail: Simon.R.Goerger@erdc.dren.mil

A.M. Madni et al. (eds.), *Disciplinary Convergence in Systems Engineering Research*, https://doi.org/10.1007/978-3-319-62217-0_1

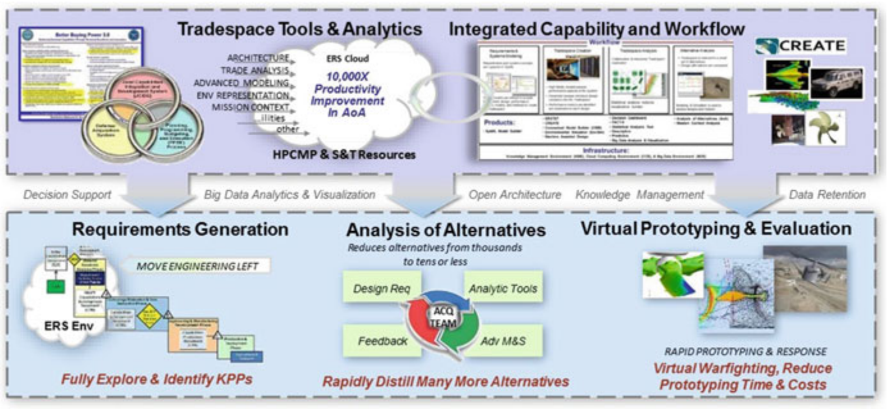

Tradespace Tools & Analytics
Integrated Capability and Workflow
CREATE
ARCHITECTURE
TRADE ANALYSIS
ADVANCED MODELING
ENV REPRESENTATION
MISSION CONTEXT
ilities
other
ERS Cloud
10,000X Productivity Improvement In AoA
HPCMP & S&T Resources
Decision Support
Big Data Analytics & Visualization
Open Architecture
Knowledge Management
Data Retention
Requirements Generation
MOVE ENGINEERING LEFT
ERS Env
Fully Explore & Identify KPPs
Analysis of Alternatives
Reduces alternatives from thousands to tens or less
Design Req
Analytic Tools
ACQ TEAM
Feedback
Adv M&S
Rapidly Distill Many More Alternatives
Virtual Prototyping & Evaluation
RAPID PROTOTYPING & RESPONSE
Virtual Warfighting, Reduce Prototyping Time & Costs

Fig. 1.1 ERS summary [1]

need to be resilient not only to planned threats and functions, but they also need to be resilient to uncertain threats and changing functionality. In the military and defense industries, current analysis of alternatives (AoA) using requirements analysis does not always plan for future threats, missions, or scenarios. However, systems cannot simply be designed for one mission; instead they need to withstand threats and have multiple functionalities. Therefore, complex systems should be engineered to be resilient to uncertain and evolving threats, missions, and scenarios.

As a response to the need for resilient systems, the Department of Defense (DoD) has created the Engineering Resilient Systems (ERS) program. ERS focuses on the effective and efficient design and development of complex resilient engineered systems throughout their life cycle. This research focuses on defining engineering resilience to enable key stakeholders such as planners, concept developers, system designers, system engineers, program managers, and system acquisition leaders to assess options to improve system resilience in the early life cycle stages. By considering resilience, the DoD strives to improve its AoA as shown in Fig. 1.1 [1]. Specifically, it seeks to improve its buying power by specifically addressing resilience early in the design cycle. In addition, it wants to add efficiency to the AoA process by using tradespace and analytics tools that use high-performance computing to explore the design space, efficiently sift through millions of designs, and quantify resilience and help analyze alternatives. Lastly, it wants to improve the design process by using Computational Research and Engineering Acquisition Tools and Environments (CREATE) that allow for virtual prototyping, design verification, and operational testing.

In order to engineer resilient systems, system designers and managers must contemplate design options considering various scenarios, missions, functions and their performance measures, threats including environmental conditions, adversary actions, detectable performance degradation, uncertain survivability, and measurable recovery over time. Resilient design options include means for flexible adaptability, which provide the ability to reconfigure and/or replace components during the system lifetime. The criteria to evaluate the design options must include the impact on performance, cost, and schedule. A tradespace analysis is critical to ensure senior decision-makers are able to determine the affordability of systems and their design options allowing for improved resilience.

With the aim of developing an appropriate framework for Engineering Resilient Systems, this paper first examines the existing academic literature. Using this literature along with stakeholder input, a set of criteria for Engineering Resilient Systems was created. To meet these criteria, a framework for incorporating resilience into AoAs was created. This framework will be used in future research to develop different methods of quantifying resilience. Lastly the literature was evaluated once again for gaps using the criteria and the framework.

1.2 Engineering Resilience Concepts and Definition

Recently our research team wrote a literature survey involving 47 papers [2]. In the research, the team found varying terms and definitions of resilience. In order to understand the literature and create a definition of resilience encompassing the varying uses and definitions of resilience, the team created the Venn diagram shown in Fig. 1.2. This Venn diagram shows the common themes and terms of resilience used in the literature search grouped into similar areas.

While designing this Venn diagram, our research led us to view resilience from two perspectives: platform and mission resilience. Platform resilience involves engineering changes and other features allowing a system platform to be flexible

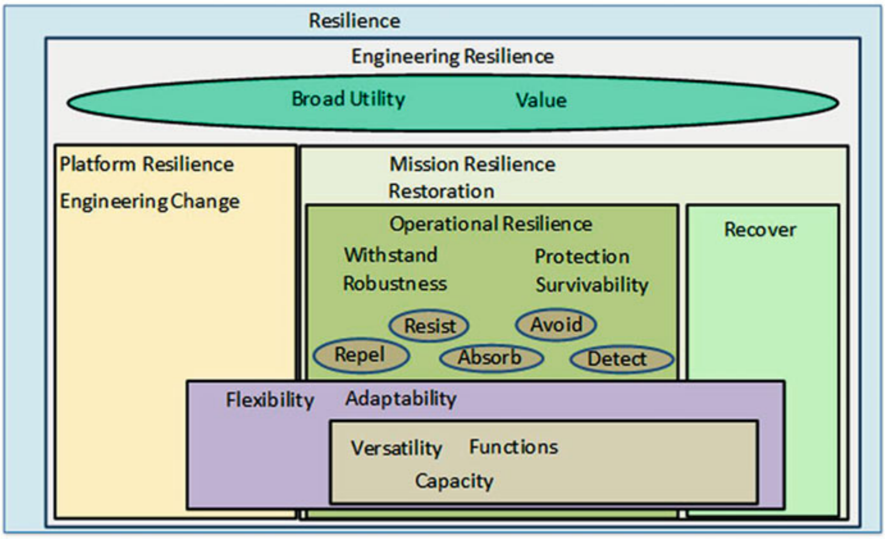


Fig. 1.2 Resilience Venn diagram [2]

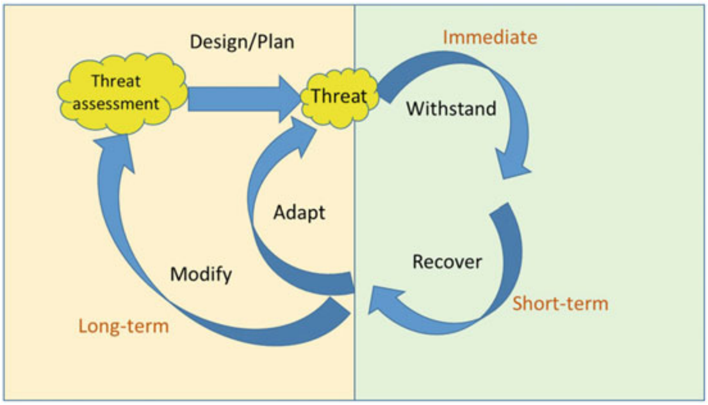


Fig. 1.3 Resilience cycle [2]

and adapt to new missions, scenarios, and threats. Mission resilience is the ability of a system to withstand and survive threats and disruptions and to recover from them quickly to achieve the mission. The majority of the literature was focused on mission resilience. Using this knowledge gained from the literature survey, the team created a broad definition of resilience:

> A resilient engineered system is able to successfully complete its planned mission(s) in the face of environmental and adversarial threats, and has capabilities allowing it to flexibly adapt to future missions with evolving threats.

Using this definition, the authors view resilience as the cycle shown in Fig. 1.3. In designing a resilient system, the process begins with a threat assessment. After this, systems are designed to face these threats. Once systems are operational and performing missions, the systems face evolving threats. Immediately they need to withstand the threats to accomplish the mission. If the system survives, it needs to recover from any damage or performance loss. After recovering, systems either return to face another threat or adapt to new threats by using platform resilience options incorporated in the original design decisions. If more significant changes are needed, the system may need to be modified. In this case, systems go through another redesign or modification process. After any redesigns, the systems face threats once more and cycle through the process until the systems are retired.

1.3 Criteria for Incorporating Resilience into Analysis of Alternatives

Using the results of the literature search, the team identified criteria a framework for incorporating resilience into AoAs. The eleven criteria are as follows: (1) use standard terms encompassing many engineering domains, (2) focus on early system definition, (3) consider multiple scenarios, (4) consider multiple threats, (5) consider short-term and long-term resilience, (6) expand the design space, (7) consider many system functions and performance measures, (8) incorporate the "Illities," (9) be independent of the modeling and simulation techniques, (10) allow for uncertainty analysis, and (11) support affordability analysis.

The framework should use common mission analysis terms from many engineering domains. A framework only using terms specific to one engineering domain cannot be easily applied elsewhere. Consequently, if it cannot be applied to other domains, it cannot be widely used or effective in general system design.

Engineering resilience must be considered early in the system life cycle. To be effective in creating resilient systems, the evaluation of engineering resilience must include the early system definition, including the "pre-Milestone A" decisions.

DoD systems are used in different missions and scenarios to perform multiple functions. Therefore, an effective framework needs to consider multiple missions and scenarios.

Numerous papers on engineering resilience focus on only one threat. Realistic DoD systems will face multiple uncertain threats throughout their life cycle. Therefore, any framework developed should allow for multiple threats rather than a single threat.

Time is a critical factor in engineering resilience. Systems face threats throughout their life cycle, and the threats may evolve or change dramatically. In addition, resilient systems not only respond and recover from threats in the short term, but they need to be able to take advantage of designed adaptability to be affordably modified to face new threats in the long term. Since resilience involves the system response to dynamic threats, a framework needs explicitly time, short-term resilience, and long-term resilience (Fig. 1.3).

The best practice is to avoid AoAs with a few point-based solutions. Point-based solutions do not provide sufficient insights about the design space. A goal of the ERS systems engineering process is to transform traditional point-based, requirements-driven design into set-based and data/analysis-driven design [3].

The majority of papers in the literature focus on one function and one performance measure. However, complex DoD systems perform several functions and have multiple performance measures.

The evaluation of resilience requires consideration of many "illities." These are terms such as availability, reliability, survivability, producibility, supportability, and others. These "illities" are a key consideration in the cost and value of systems. Hence, they need to be considered in a resilience framework.

Mission analyses and AoA use modeling and simulation techniques tailored to the system and the availability of data. Since modeling is the best way to estimate cost and value in early life cycle stages, the framework must be independent of the modeling and simulation techniques used in AoA.

Uncertainty is a reality in engineering resilience decisions. Many DoD systems have service lives lasting for decades. During this time, missions, scenarios, and threats change as new technology and adversaries arise. In addition, every situation is different and can have different outcomes leading to uncertain performance and cost. Therefore, the framework should explicitly consider uncertainty.

Affordability is an important consideration in system development. "Big A" affordability evaluates and assesses the value versus the costs at major milestones. "Little a" affordability refers to the continual evaluation of the value versus cost on all program decisions. Since the DoD and all decisions makers are concerned with cost and value, the framework must support affordability analysis.

In summary, these criteria together will allow for incorporating resilience into analysis of alternatives. However, as a list, these criteria mix both best practices and ERS requirements. To resolve this, Fig. 1.4 includes the criteria, the flow of time, and as a result the dependencies. In addition, it identifies the new steps to analysis of alternatives that ERS adds in red lettering. In addition, they do not show the sequence and dependencies involved in the criteria. The sequence and dependencies are shown in Fig. 1.5.

Analysis needs to begin with identifying missions, scenarios, and value gaps. Next, ERS adds a step to AoA requiring expanding the design space and providing resilience options. Then cost drivers, performance measures, and relevant illities need to be determined. To quantify the uncertainty in these, engineers need to perform modeling and simulation. Using these, the value tradespace and the costs need to be quantified. In addition, during this stage, another step is added in AoA to extend the service lifetime. Specifically, this will analyze the effects of platform resilience and responds to evolving threats and scenarios. Using the previous analysis, decision-makers can make resilience and affordability trade-offs. In

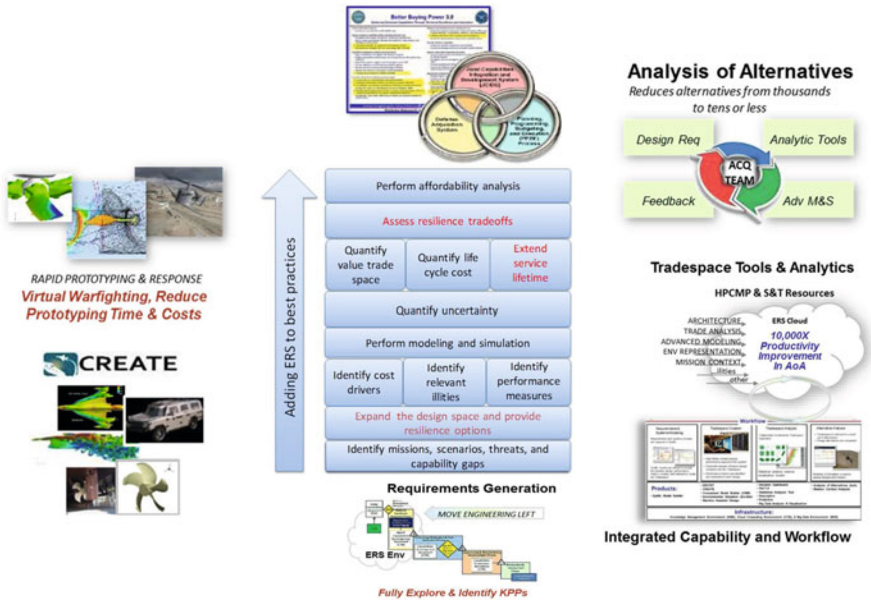

RAPID PROTOTYPING & RESPONSE
Virtual Warfighting, Reduce Prototyping Time & Costs
CREATE
Adding ERS to best practices
Perform affordability analysis
Assess resilience tradeoffs
Quantify value trade space
Quantify life cycle cost
Extend service lifetime
Quantify uncertainty
Perform modeling and simulation
Identify cost drivers
Identify relevant illities
Identify performance measures
Expand the design space and provide resilience options
Identify missions, scenarios, threats, and capability gaps
Requirements Generation
MOVE ENGINEERING LEFT
ERS Env
Fully Explore & Identify KPPs
Analysis of Alternatives
Reduces alternatives from thousands to tens or less
Design Req
Analytic Tools
ACQ TEAM
Feedback
Adv M&S
Tradespace Tools & Analytics
HPCMP & S&T Resources
ARCHITECTURE
TRADE ANALYSIS
ADVANCED MODELING
ENV REPRESENTATION
MISSION CONTEXT
illities
other
ERS Cloud
10,000X Productivity Improvement in AoA
Integrated Capability and Workflow

Fig. 1.4 Incorporating ERS into analysis of alternatives

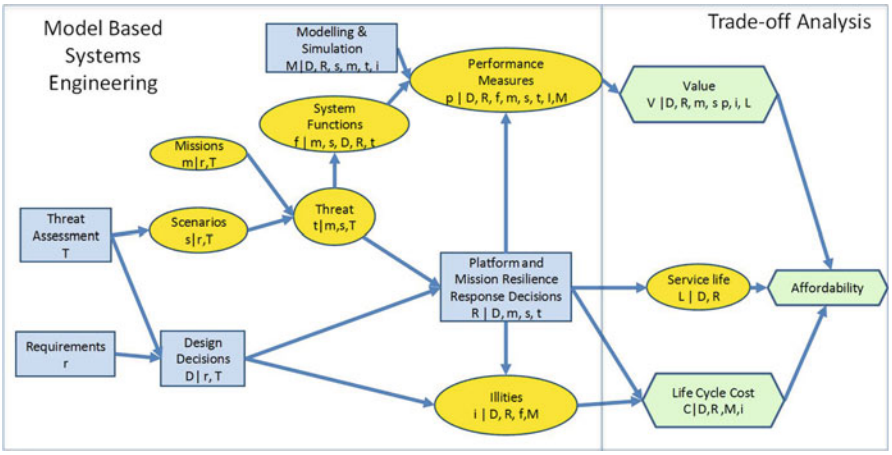

Model Based Systems Engineering
Trade-off Analysis
Modelling & Simulation M|D, R, s, m, t, i
Performance Measures p | D, R, f, m, s, t, I,M
System Functions f | m, s, D, R, t
Missions m|r,T
Threat Assessment T
Scenarios s|r,T
Threat t|m,s,T
Value V |D, R, m, s p, i, L
Platform and Mission Resilience Response Decisions R | D, m, s, t
Service life L | D, R
Affordability
Requirements r
Design Decisions D| r, T
Illities i | D, R, f,M
Life Cycle Cost C|D,R,M,i

Fig. 1.5 Framework for incorporating ERS into analysis of alternatives

addition, these criteria incorporate the goals and objectives of ERS. The CREATE and tradespace tools will be used to expand the design space and modeling and simulation to provide detail required to better predict costs and values. Lastly, affordability analysis and resilience trade-offs in the AOA directly support the better buying directives of DoD.

1.4 Proposed Framework for Incorporating ERS into Analysis of Alternatives

In order to help make decisions during the early life stages of systems, the authors created a framework to incorporate the criteria into analysis of alternatives. Visually this framework is shown as an influence diagram in Fig. 1.5. An influence diagram is a concise representation of a decision problem or opportunity [4]. They identify the variables and their relationships but suppress the details. They use four nodes: decision nodes, uncertainty nodes, constant nodes, and value nodes. A decision node signifies the decision alternatives or options and is displayed by a rectangle. An uncertainty node represents the different outcomes of an uncertain event and is depicted as an oval. A constant node symbolizes a function or number that will not change and is depicted by a diamond shape. Lastly, an influence diagram has value nodes denoting the decision-makers' preferences for outcomes. Value nodes can have different types of values such as cost, performance measures, or an affordability based on cost, performance, and service life. A hexagon depicts a value node. In the diagrams, arrows are used to display influences. There are two types of influences: a probability relationship and the availability of information. The time sequence of the events is from left to right.

In Fig. 1.5 the nodes are:

- Threat assessment, T – a decision that identifies the anticipated threats the system will face.
- Requirements, r – a set of decisions determining the use of the system and required minimum performance.
- Design decisions, D – a set of decisions made with knowledge of the requirements and threat assessment. These can be point-based design decisions or set-based decisions. However, only set-based design decisions will meet the requirements to expand the design space.
- Modeling and simulation, M – the decisions made which methods and techniques used to model and what scenarios and missions to simulate the system in order to predict measures, illities, and costs.
- Platform and mission resilience response decisions, R – a decision node representing mission response decisions (short term) and platform response decisions (long term) informed by threats.
- Scenarios, s – a chance node representing an uncertain scenario, which may or may not be in the original threat assessment or requirements analysis.
- Missions, m – a chance node representing the missions the system is actually used on; this may or may not be included in the initial threat assessment or requirements analysis.
- Threat, t – a chance node representing the uncertain threat that depends on the mission. There can be different threats to different system functions. In this diagram, threat is the term used for any adverse event (environmental or adversary) that could degrade any capability of the system. This may or may not be in the original T.
- System functions, f – a chance node determining how the system is used; it is influenced by the missions and scenarios the system is used in.
- Performance measures, p – a chance node depending on the function, the illities, modeling and simulation, and resilience response decisions.
- Illities, i – a set of chances such as reliability, survivability, availability, and others affecting the performance and cost of the system.
- Service life, L – a chance node affected by the performance of the system, the illities, and the resilience response decisions.
- Value, V – a value node depending on the performance for the mission for all functions and several other variables.
- Life cycle cost, C – a value node depending on the design, the producibility, the supportability, and the platform and mission response decisions.
- Affordability, A – a value node comparing value versus life cycle cost.

In a defense design process, intelligence analysts first determine what threats new systems will face in the threat assessment. In addition, requirements for the system will be analyzed. The threat assessment and requirements analysis inform the design decisions. After the design decisions, the systems are employed in uncertain missions and scenarios. Next, even though there is an initial threat assessment, the threats the system faces are uncertain and based on the scenarios and missions the system is sent to perform. When facing these threats, users can make resilience response decisions based on uncertain threats. These options are based on what the design decisions allow the system to do and how they allow it to adapt. These decisions can be short term such as avoiding a threat. They can be a simple modification of the system like adding a new sensor. Or they can also be long-term decisions to significantly modify or adapt to the system to face new threats. For instance, a historical long-term decision was to add weapons to a C-130 to change the functionality of the C-130 from tactical airlift to a gunship. Depending on the threats, design decisions, and response decisions, the system can have different functions. From these functions, the illities, or set of characteristics including reliability, availability, producibility, survivability, etc., are determined. Next, the functions, threats, scenarios, missions, illities, design decisions, and the resilience response decisions influence the performance. The performance measures should be estimated using modeling and simulation. But, the choice of specific models and simulations is a decision because analysts need to decide which type of models and simulations to use for each system. From the performance, the value of the system is determined. Using the design decisions, the resilience response decisions, and the illities, the life cycle cost is evaluated. Then the service life is estimated from the response decisions, the design decisions, and the performance. Using the estimated values, costs, and life cycles, decision-makers make affordability decisions early in the life cycle.

Throughout this decision process, the uncertainties should be estimated using model-based systems engineering. Model-based systems engineering is a process of engineering systems using modeling throughout the AoA and decision process. Various types of models should be included. Specifically, the systems will each need at least a physics-based performance model and detailed cost model.

In addition, using the data from the right side of the framework, decision-makers need to perform trade-offs between value and cost. To balance the needs for high-performance systems and with the budget requirements, the decision-makers need data on the life cycle length, the life cycle costs, and the value of the system to allow them to assess the affordability of the system.

This framework for resilience was created to fit the criteria for incorporating resilience into AoAs. Although many of the terms are drawn from the defense industry, the framework as a whole can be applied to many different areas. In particular, the authors are currently applying the problem definition to two systems with application in the defense industry and in the public sector: unmanned aerial vehicles (UAVs) and autonomous vehicles. The analysis will focus on the early-stage decisions. It allows for set-based design. Both short-term and long-term resilience are considered in the platform and mission resilience response decisions. Multiple scenarios, threats, functions, and performance measures can be considered. The illities are incorporated. In addition, leaving the chance node as the broad term "illities" allows for inclusion of any illities a system might be concerned with. This framework accounts for uncertainty. And many different types of modeling and simulation can be used with this framework. Lastly, the framework enables affordability analysis.

1.5 Comparing the Literature to the Framework

Lastly multiple papers in the literature were analyzed using the criteria and the framework. In Fig. 1.6, the green boxes show papers fully meeting each criterion, red illustrates where papers fell short on each criterion, and the yellow displays where papers partially meet, but can be improved on each criterion. Lastly, the papers in the figure are organized from top to bottom based on first how many criteria they met and second how many they partially met.

Out of 13 papers:

- 5 papers consider resilience in general systems.
- 4 papers consider resilience in the early stages.
- 1 paper considered expanding the design space using set-based design.
- 8 papers considered both short- and long-term resilience.
- 1 paper considered multiple scenarios.
- 6 papers considered multiple threats.
- 5 papers considered multiple functions and performance measures.
- 10 papers considered the "illities."
- 5 papers used uncertainty analysis.
- 7 papers used modeling and simulation techniques.
- 3 papers supported affordability analysis.

Moreover, although meeting many of the criteria, no papers met all of the criteria required for the framework. Therefore, since many of the criteria are not considered by a large number of papers and no single paper met all of the criteria, there are gaps in the literature.

1.6 Future Work

The authors have five activities planned for future work. First, we will continue to present this work at various conferences. This allows feedback to improve the definition of engineering resilience, the engineering resilience cycle, and the framework for incorporating resilience into AoAs. Second, we will continue the literature search to identify possible solutions to the identified gaps and refine the framework. Third, we will validate the framework using illustrative engineering examples including autonomous systems (e.g., UAVs and autonomous vehicles). Fourth, the framework will be expanded to account for manned and cyber systems. Lastly, the team is researching different methods of resilience quantification to use with the framework.

	Framework Criteria														
	Criteria	Use terms encompassing many engineering domains	Focus on the early system definition	Expands Design Space	Consider short and long term resilience		Considers multiple scenarios	Considers multiple threats.	Considers system functions and performance measures		Incorporate the "Illities"	Allow for Uncertainty analysis	Be independent of the modeling and simulation techniques	Support affordability analysis	
	Elements	*Engineering domain*	*Life cycle stage*	*Discusses Set Based Design or other design space expansion*	*Short Term*	*Long Term*	*# of scenarios explicity considered*	*Threats*	*Systems functions*	*System performance measures*	*Number of "Illities"*	*Techniques used*	*M & S techniques*	*Value-multiple/ single*	*Cost*
Papers	Sitterle et al. [5]	General DoD sytstems	Concept Stage	Discusses Set Based Design	Yes	Yes	0	Not explictiy discussed- Allows for consideration of multiple	Multiple	Multiple-Utility	Flexibility, chageability, adaptability, agility, versatility	Discusses	None	Multiple	Yes
	Youn et al. [6]	General Engineered Systems	Early Stage	No	Yes	Yes	1	Internal threats- resilience a subset of reliability	Multiple	Single-Cost	Reliability, flexibility, adaptability, detectibility	Prognostics and Health Mangagent methods	Prognostics and Health Mangagent methods	None	Yes
	Ross et al. [7]	Aerospace Systems	Design Stage	Design Vectors	Yes	No	0	Environmental Disturbances	Single Function	Multiple utility measures for one function	Adaptability, flexibility, changeability, survivabililty.	Monte Cartlo Simulations	Physics Based Models	Multiple Utility Meausres	Yes
	Shafieezadeh and Burden. [8]	Seaport Operations	Existing systems	No	Yes	No	1	Seismic Disturbances	Multiple	Multiple	Fragility, vulnerability, reliability, flexibility, availability (of equipment)	Ground Motion Prediction Equations	Ground Motion Prediction Equations	Multiple	Yes
	Kahan et al. [9, 10]	Infrustructure and Buildings	Early Stage	No	Yes	Yes	1	Terrorist Attacks	Multiple	Multiple	Flexibility, broad applicability, vulnerability, adaptability	Discusses	None	Single: resilience.	Yes
	Henderson and Lancaster [11]	Technology Enterprise	Existing systems	No	Yes	Yes	Multiple but not identified	Multiple but not identified	Multiple	Multiple	recoverability, vulnerability, availability	Discusses	OODA control Loop	Multiple	Included in performance metrics
	Jackson and Ferris [12]	Organizational, physical, procedural systems	No specific lifecycle stage	Resilience principles (ie Redundancy, etc)	Yes	Limite d	0	Terrorist attacks, Natural Disasters, Human and design errors.	Multiple	Multiple	Adaptability, vulnerability, survivablity, flexibility, reliability, repairability	Discusses	None	Principles of resilience not Value	No
	Ettouney [13]	Communities and Civil Infrastructure	Existing systems	No	Yes	Yes	1	Man Made events Natural Events	Single: survive	Multiple to measure Resilience	Sustainability, vulnerability, reliability, availability (of workforce)	US-DHS pubically avaiblable tools- compute resilience for buildings, trasit stations and tunnels.	US-DHS pubically avaiblable tools- compute resilience for buildings, trasit stations and tunnels.	Single: Resilience	Yes
	Madni and Jackson [14]	General Systems	Throughout Lifecycle	No	Yes	Yes	1 for each system	External: environmental, adversary; Systemic: Internal threats, human errors	Single: resilience- Framework designed to measure resilience	Multiple to measure Resilience	Adaptability, flexibility, reliability, survivability, changeability, predicability, inspectability	Discusses	None	Single currently, could be included in utility	Yes
	Sheard et al. [15]	General Systems	Existing systems	Resilience principles (ie Redundancy, etc)	Yes	Yes	0	Failures, Accidents, Attacks, Pathogesn, Natural Hazards, Competetive Challenge	Single: survival	Single: resilience	Survivability, reliability, agility, flexibility, vulnerabilty	Discusses	None	None	None
	National Academies [16]	Communities and Civil Infrastructure	Existing systems	No	Yes	Yes	Mulltiple	Natural Disasters and Man Made Disasters-i.e. Terrorism, financial crises, or social unrest	Single -Survive/ recover	Multiple to measure Resilience	Vulnerability, flexibility, recoverability,	None	San Fransisco Planning and Urban Resarch Association (SPUR) Method	Single: resilience	No
	Franchin and Cavalieri [17]	Civil Infrastructure	Existing systems	No	Yes	No	1	Earthquakes	Single: survival	Multiple to measure Resilience	Vulnerability, reliability, availability (of tools), flexbility, adapability	Network of random variables	Using network models	Single: resilience.	No
	Bruneau et al. [18]	Physical Communities	Existing systems	Resilience principles (ie Redundancy, etc)	Yes	No	1 for each type of system	Seismic Disturbances	Single: Survive	Multiple to measure Resilience	Sustainability, flexibility, reliability, fragility, vulnerability, availability, viability	Discusses	None	Single: resilience.	Limited

Fig. 1.6 Literature conceptual framework analysis

1.7 Conclusion

The authors propose a definition of resilience that is independent of the means to achieve resilience. Using this definition alongside the research from the literature search and knowledge from stakeholders, the team identified criteria for incorporating resilience into AoA. These efforts mix AoA best practices and ERS. Figure 1.4 identifies best practices and ERS in addition to showing how the criteria fit into ERS. Using this criterion, the team has created a framework to incorporate ERS into AoA and to quantify resilience (Fig. 1.5). This framework is represented as an influence diagram that can be used by many different modeling techniques. In addition, the framework fulfills the criteria identified. In the future, the framework will be revised and refined through peer reviews. In addition, the framework is currently beginning to be applied to two different autonomous applications. Using the framework, the team will continue its research and develop methods of quantifying resilience to fill the identified research gaps.

References

1. Holland JP (2015) Engineered Resilient Systems: Power of advanced modeling and analytics in support of acquisition. NDIA 16th Science and Engineering Technology Conference, Springfield
2. Cottam B, Parnell G, Pohl E, Specking E, Small C (2016) Quantifying resilience to enable Engineered Resilient Systems: Task 1 Report. CELDI, Fayetteville
3. Parnell, G., Goerger, S., Pohl, E. "Reimagining Tradespace Definition and Exploration, "Proceedings of the American Society for Engineering Management 2017 International Annual Conference, 18-21 Oct 2017, E-H. Ng, B. Nepal, and E. Schott eds
4. Parnell GS, Bresnick TA, Tani SN, Johnson ER (2013) Handbook of Decision Analysis. John Wiley & Sons, Inc., Hoboken
5. Sitterle V, Freeman D, Goerger S, Ender T (2015) Systems engineering resiliency: guiding Tradespace exploration within an engineered resilient systems context. Procedia Comput Sci 44:649–658. Elsevier
6. Youn B, Hu C, Wang P (2011) Resilience-driven system design of complex engineered systems. Am Soc Mech Eng 133(10):101011
7. Ross A, Stein D, Hastings D (2014) Multi-attribute tradespace exploration for survivability. AIAA J 51(5):1735–1752
8. Shafieezadeh A, Burden L (2014) Scenario-based resilience assessment framework for critical infrastructure systems: case study for seismic resilience of seaports. Reliab Eng Syst Saf 132:207–219. Elsevier
9. Kahan JH, Allen AC, George J (2009) An operational framework for resilience. J Homeland Sec Emer Manag 6(1), Article 83.
10. Kahan J, Allen A, George J, Thompson G (2009) Concept development: an operational framework for resilience. Homeland Security Studies and Analysis Institute, Arlington VA
11. Henderson D, Lancaster M (2015) Value modeling for enterprise resilience. INCOSE Int Symp 25(1):1417–1426
12. Jackson S, Ferris T (2013) Resilience principles for engineered systems. Syst Eng 16 (2):152–164. Wiley Online Library

13. Ettouney M (2014) Resilience management: How it is becoming essential to civil infrastructure recovery. McGraw Hill Financial, New York City
14. Madni A, Jackson S (2009) Towards a conceptual framework for resilience engineering. IEEE Syst J 3(2):181–191
15. Sheard S, Mostashari A (2008) A framework for system resilience discussions. INCOSE International Symposium (Vol. 18, No. 1, pp. 1243-1257).
16. The National Academies (2013) Disaster resilience: a national imperative: summary. The National Academies Press, Washington, DC
17. Franchin P, Cavalieri F (2015) Probabilistic assessment of civil infrastructure resilience to earthquakes. Comput Aided Civ Inf Eng 30(7):583–600. Wiley Online Library
18. Bruneau M, Chang S, Eguchi R, Lee G, O'Rourke T, Reinhorn A et al (2003) A framework to quantitatively assess and enhance the seismic resilience of communities. Earthquake Spectra 19(4):733–752

Chapter 2
Early Life Cycle Cost Estimation: Fiscal Stewardship with Engineered Resilient Systems

Travis Moody, Robert Provine, Samantha Todd, Nicholas Tyler, Thomas R. Ryan, and Ricardo Valerdi

Abstract Organizations are constantly seeking to achieve earlier and more accurate cost estimates in order to make better trades space and design decisions, as well as minimize project cost and schedule overrun. These estimates facilitate decisions that are more informed – especially within the United States Department of Defense's engineered resilient systems (ERS) program. This paper will discuss the current methods used to achieve life cycle estimates, the role of estimation within ERS, and recommend a parametric life cycle cost estimation model that will support decision-making. In addition, this paper will focus solely on early life cycle engineering inputs that translate with Department of Defense's pre-Milestone A in order to create an early life cycle cost estimation model (ELCE). This model leverages the engineering inputs (design parameters) that are typically available early in the design process in the following five categories: hardware, software, systems engineering, project management, and integration. This paper will also highlight future research goals to determine values for factors of economies of scale, regression analysis with real data, limitations, and potential impacts of application.

Keywords Acquisition • DoD 5000.02 • Engineered resilient systems • Life cycle • Costing • Hardware estimation • Software estimation • Systems engineering estimation • Project management • Project management estimation • Integration • Integration estimation • COSYSMO • COCOMO II • Cost estimation relationships

T. Moody (✉) • R. Provine • S. Todd • N. Tyler • T.R. Ryan
United States Military Academy, West Point, NY, USA
e-mail: travis.moody@usma.edu

R. Valerdi
University of Arizona, Tucson, AZ, USA
e-mail: rvalerdi@arizona.edu

A.M. Madni et al. (eds.), *Disciplinary Convergence in Systems Engineering Research*, https://doi.org/10.1007/978-3-319-62217-0_2

2.1 Introduction

Estimating the life cycle cost of a new system, in a stage early enough to procure funding, is a difficult proposition. The Department of Defense (DoD) is at the forefront of military costing, but the institution as a whole needs to produce estimates that are more effective. For example, when the F-35 Lightning II Program was proposed for funding in October 2001, the total program cost was estimated to be $224.77 billion dollars for 2866 units. As of August 2013 after 121 months behind schedule, the total program cost had soared to $332.32 billion dollars. This constitutes an increase of 47% while producing 409 less units and a 72% increase in per unit cost from the original estimates [9]. This is unacceptable and breaches the trust between the citizens of the United States, the government, and the DoD [4, 5].

Engineered resilient systems (ERS) is one such DoD program attempting to help reduce cost-associated problems by attaching life cycle estimates to decision alternatives. ERS is housed within the US Army Engineer Research and Development Center (ERDC) and aims to provide a data-driven approach to building resilient systems through trade space analysis tools [10]. Within the ERS suite of tools, there exists a need for an embedded cost estimation component that is provided as an output to ERS users during the early stages (pre-Milestone A) of a system life cycle, as shown in Fig. 2.1. As design parameters are entered into the ERS tradespace tool, a SysML-like architecture is created, which allows for the generation of life cycle cost estimates. These estimates will then be attached to different design alternatives to aid engineers in the decision-making process.

Established methods for determining the cost of a system are described as top-down, bottom-up, and parametric [1, 15]. This research is concerned

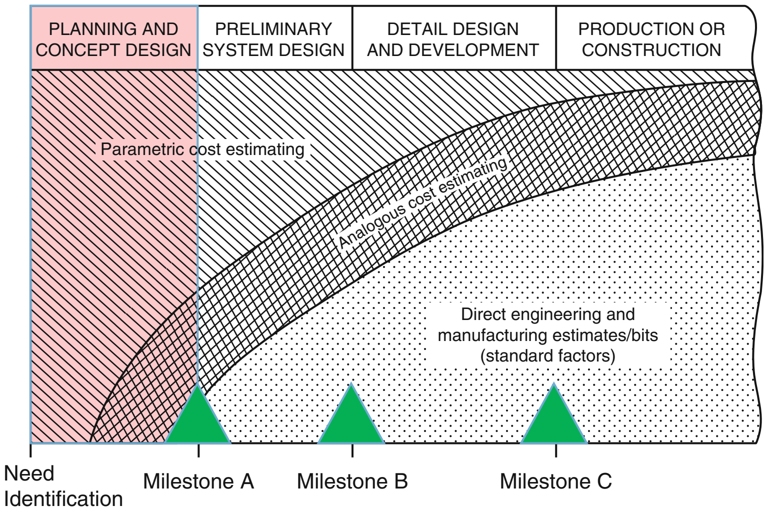

PLANNING AND CONCEPT DESIGN
PRELIMINARY SYSTEM DESIGN
DETAIL DESIGN AND DEVELOPMENT
PRODUCTION OR CONSTRUCTION
Parametric cost estimating
Analogous cost estimating
Direct engineering and manufacturing estimates/bits (standard factors)
Need Identification
Milestone A
Milestone B
Milestone C

Fig. 2.1 DoD acquisition milestones overlaid on general cost estimating techniques by engineering phase. This depicts the three milestones of the DoD acquisitions process in relation to the systems phases. The *red* box outlines the boundary of the research in this paper [1, 6]

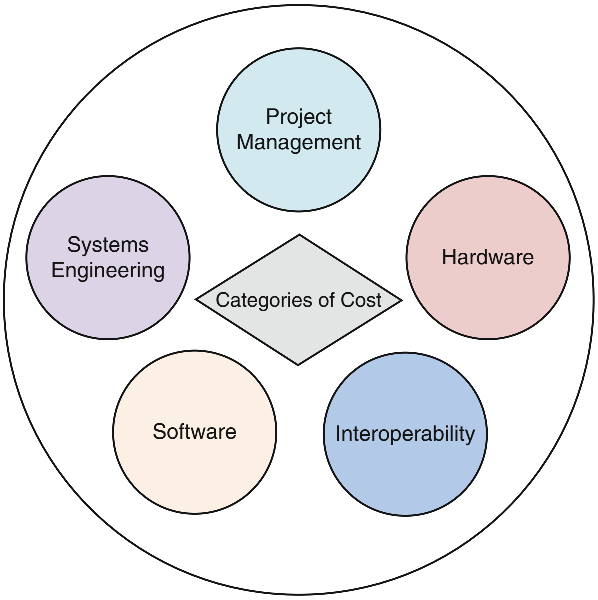


Fig. 2.2 Five categories of system level costs. The inputs for a parametric model are derived from more general areas of cost. The areas of cost identified in this research are software, hardware, project management, systems engineering, and integration. Parametric models currently exist for each area except integration [2, 13, 15, 17]

particularly with pre-Milestone A (pre-MS A), a program review between the conceptual and preliminary design phase in the DoD 5000.02 acquisition process. Referring to Fig. 2.1, most cost decisions are made with parametric models or a more subjective top-down method pre-MS A. Parametric cost estimation uses inputs, also called parameters, to help shape the mathematical relationships that will produce a more accurate estimate [15]. Currently there is no complete parametric model that accounts for total system costs; the existing models are discrete cost categories. This research is aimed at developing a parametric model that can be used to develop a total system cost pre-MS A.

When estimating the total costs of a system, it is intuitive to consider factors that specifically drive the cost of software, hardware, project management, systems engineering, and integration, as shown in Fig. 2.2. This exists because most systems are made up of some combination of the five categories [7]. The goal of this paper is to introduce a comprehensive parametric model which utilizes inputs that are available early in a system's life cycle, already contained within pre-existing models, that can be used to generate an effort output associated with cost. The aggregate of these costs will enable ERS users to make better tradespace decisions.

2.2 Background

In the current paradigm of cost estimation, there exists different estimation models specifically for the various types of systems; however, those models remain limited to their corresponding areas of cost, which therefore limits cost estimators' ability

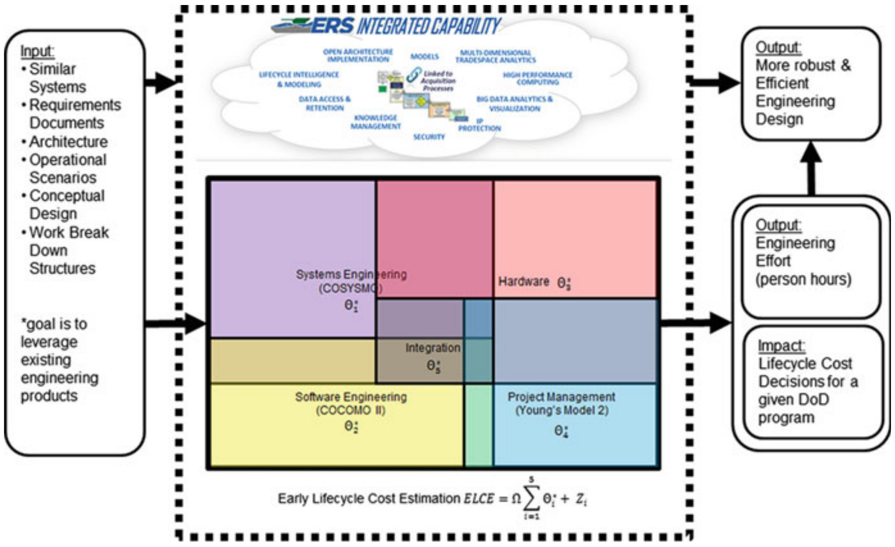

Input:
• Similar Systems
• Requirements Documents
• Architecture
• Operational Scenarios
• Conceptual Design
• Work Break Down Structures
*goal is to leverage existing engineering products
ERS INTEGRATED CAPABILITY
OPEN ARCHITECTURE IMPLEMENTATION
MODELS
MULTI-DIMENSIONAL TRADESPACE ANALYTICS
LIFECYCLE INTELLIGENCE & MODELING
Linked to Acquisition Processes
HIGH PERFORMANCE COMPUTING
DATA ACCESS & RETENTION
BIG DATA ANALYTICS & VISUALIZATION
KNOWLEDGE MANAGEMENT
IP PROTECTION
SECURITY
Systems Engineering (COSYSMO) θ_1
Hardware θ_2
Integration θ_3
Software Engineering (COCOMO II) θ_5
Project Management (Young's Model 2) θ_4
Early Lifecycle Cost Estimation $ELCE = \Omega \sum_{i=1}^{5} \theta_i + Z_i$
Output: More robust & Efficient Engineering Design
Output: Engineering Effort (person hours)
Impact: Lifecycle Cost Decisions for a given DoD program

Fig. 2.3 The graphic depicts the five categories of cost, with their perceived overlaps, and the interaction between engineering inputs and ERS outputs [2, 3, 8, 10, 13, 15, 17]

to generate total life cycle cost estimates [2, 13, 15]. The models being leveraged to account for the four of the identified cost categories are constructive cost model (COCOMO II), the constructive systems engineering cost model (COSYSMO), Young's model 2 for project management, and SEER hardware (SEER-H). Currently, we are exploring that the integration component can be generated by the overlap of the other four categories of cost, as depicted in Fig. 2.3. Each of these cost models takes into account many different engineering inputs, which shape a system's cost, but are limited in that those inputs are for the whole life cycle and assume that everything about the system is known. The pre-existing models, for the most part, allow cost estimators to generate a life cycle cost estimate with what is known as pre-MS A. The goal of this research is to identify the parameters that are *known* early in the design tradespace process, for those existing models, and then combine them to create a total life cycle cost estimate. This is the fundamental difference between simply collecting cost estimates for the five cost categories and creating a parametric cost model which accounts for limited parameter availability [16]. Finally, as opposed to estimating for a specific system, this model aims to be more encompassing. A holistic cost model will provide a more complete picture of life cycle cost and will shed light on the interactions between cost elements which will increase its applicability within ERS.

2.3 Methodology

The methodology used by Valerdi (2008) in the development of COSYSMO leveraged a seven step process for building parametric models similar to what Boehm, et al. (2000) utilized in developing COCOMO II [2, 15]. This research will adapt some of those steps beginning with analyzing existing literature to understand the factors that influence life cycle cost. After reviewing the literature, next in the process is to leverage experts within the cost estimation field to refine any insight gained with research. Finally, this paper will explore how to model the concepts which will lead to the creation of an early parametric life cycle cost model.

Discussion with experts suggests that life cycle cost can be summarized into the five categories identified earlier in the paper [7]. The first step is to identify existing models that can be leveraged for each of these five categories of cost. The second step is to proceed with identifying which of the engineering inputs in these models will fit best early in the life cycle. The third step is to identify a method to account for cost categories that are not able to be represented by an existing parametric model. The fourth and final step is to analyze the model for integration. This step is necessary to determine overlap, if any, between parameters of the five individual categories of cost.

2.4 Preliminary Results and Analysis

Based on the methodology described above, the inputs available early in the life cycle were determined to focus the development of the initial early life cycle costing parametric model. This will allow engineers and decision makers to generate a parametric cost estimate earlier in the acquisition process. ERS with the added costing component enables decisions based off of differentiation of alternatives relative to this model.

2.4.1 Initial Model

The proposed cost estimating relationship (CER) for the early life cycle cost estimation (ELCE) model includes a combination of discrete cost models for each of the cost categories, referenced in Fig. 2.1. Current practice of creating parametric models dictates that the life cycle cost can be calculated by adding the outputs of the individual models and adjusting the CER with a scale factor as follows:

$$\text{ELCE} = \text{Scale Factor}^{*}\big(\text{COCOMO II} + \text{COSYSMO} + \text{SEER-H} + \text{PM} + \text{Integration} \tag{2.1}$$

$$\text{ELCE} = \Omega \sum_{i=1}^{5} \Theta_i^{*} + Z_i \tag{2.2}$$

Θ_1* = COCOMO II with factors only ascertainable post milestone A suppressed
Θ_2* = COSYSMO with factors only ascertainable post milestone A suppressed
Θ_3* = SEER-H with factors only ascertainable post milestone A suppressed
Θ_4* = Integration factor with factors only ascertainable post milestone A suppressed
Θ_5* = Young's model 2 with factors only ascertainable post milestone A suppressed
Z_i = Average cost of suppressed factors by model Θi
Ω = Model adjustment factor based on product line historical data heuristics

2.4.2 Inputs

Determining the inputs of each cost component that are available early on is important because the separate models assume everything is known about a system [2, 15]. Leading up to pre-Milestone A, relatively little is known about a system. It is for this reason that we must identify the parameters that are available to cost estimators.

The category-specific cost models we will use are COCOMO II (software), COSYSMO (systems engineering), SEER-H (hardware), and Young's project management model 2. Possible methods to account for integration costs include determining the function points of a system or utilizing the parameters identified by Ford and Shibata [8, 14]. Looking at the engineering inputs that feed into each model, we determined which inputs are identifiable early on in the life cycle similar to the COCOMO II early design model [2].

COSYSMO factors available pre-MS A include the size drivers of *number of requirements* and *number of operational scenarios*, both rated on a scale of easy, nominal, or difficult. The cost drivers that can be derived early in the life cycle are *stakeholder team cohesion*, *personnel/team capability*, *personnel experience/continuity*, and *technology risk* [15]. Each cost drivers rated on a scale that ranges from very low, low, nominal, high, to very high [15].

Project management inputs that are identifiable early in the life cycle and are independent of systems engineering drivers were derived from Young's model 2. The effort factor parameters applicable to early estimations are *requirements* and *scope*, *project complexity* and *risk*, *project constraint*, *stakeholder cohesion* and *multisite coordination*, and *documentation* and *communication level*. The efficiency multipliers, identifiable early in the life cycle, for the above factors are *people capability*, *process maturity*, and *tool support* [17].

COCOMO II has three parameters that can be estimated early in the systems life cycle. They are project size in terms of *source lines of code (SLOC)*, *number of function points*, and the *adaptability* and *reuse of the system* [2].

For hardware costs, this model will utilize a cost estimation relationship based on a few high-level parameters within the SEER-H suite that are related to hardware costs. These inputs are *operating environment*, *material composition*, *certification level*, *classification*, and *developer capability and experience* [13, 15]. They are ascertainable early in the life cycle due to their conceptual nature and provide a starting point for formulating a hardware cost estimate.

There is an absence of a pre-existing model for the integration cost of a system. There is limited research on this subject; however, Ford has generated some research that there are approximately six measures of effectiveness, which can impact the integration cost of a system. These measures are personnel requirements, level of systems cost, functional area performance, supporting-to-supported ratio, reconstitution capability, and satisfaction of the organization's priorities [8]. The inputs that are likely available early in a system's acquisition process are *personnel requirements*, *the level or magnitude of the systems cost*, *the systems anticipated functional area performance*, and *the ratio of supporting systems to supported systems*. We also anticipate that the number of function points is an additional cost driver. It is inherent that as the number of function points for a system increases, so does the subsequent cost of integration. The inputs suggested by Ford in conjunction with the number of function points of the system will help to provide a more accurate integration cost estimate [8].

2.5 Discussion

The nature of the research causes the creation of a robust parametric model and identification of specific engineering inputs by pre-MS A to be quite challenging. Several limitations to the research exist which add to the complexity of creating a parametric cost model that is robust and accurate enough for the ERS suite of tools. When creating this parametric model, it was identified that there should be an adjustment factor included for each category of cost. These factors would depend on how prevalent that category of cost is within the overall system's life cycle cost, the availability of historical project data, and the amount of uncertainty that specific category of cost experiences.

Within the above models, there exists overlap for multiple inputs, especially those related to systems engineering. For example, function points or system requirements might be present in multiple portions of the model. Such overlap can be problematic because it can lead to double counting, which results in an inflated estimate. However, repetition in inputs may not necessarily cause overlap in the model because they account for different sources of effort for the system. It will be necessary to determine a method to account for this redundancy in the model similar to how Wang, et al. decoupled overlaps between software and systems engineering cost drivers [18].

If a cost input is accounted for in multiple categories, then the decision must be made about which category that cost actually belongs in. The same inputs can represent separate levels of work in different categories. For example, a project manager and systems engineer might be working on the same portion of a system but generate separate efforts, each with distinct costs. This perceived overlap could create a more accurate cost estimate, as it might account for gaps in the model.

The ELCE model, both in its conceptual and mathematical forms, is the first step in developing a parametric life cycle cost estimate which can be integrated into ERS's tradespace analysis tool. To continue research and development of this model, we will collect data about different systems to validate the model. Subsequently, we will apply the parametric model to the data in order to test its accuracy. This will also help to determine the robustness of the model across different types of systems. Once the model has been tested, it will be adjusted for any identified inaccuracies. This process will be repeated until the model reaches an acceptable level of confidence.

2.6 Conclusion

The research described in this paper aims to build a comprehensive parametric model that takes the inputs from multiple cost paradigms and generates a life cycle estimate. By leveraging the inputs available early in a system's life cycle, we aim to provide ERDC ERS with a pre-MS A cost estimation tool. Once fully developed, this tool can be integrated into ERS's high-powered computing tradespace analysis platform in support of the DoD acquisitions process. This research will assist ERS users to perform better analysis and comparison of alternatives. This research will be applied to data on existing DoD systems and analyzed against their actual, completed, life cycle costs in order to refine and calibrate the model. By contributing to the tradespace analysis tool, this research will provide a cost estimation method other than current DoD acquisitions methods.

References

1. Blanchard BS, Fabrycky WJ (2011) Chapter 17: Design for affordability (life cycle costing). In: Fabrycky WJ (ed) Systems engineering and analysis, 5th edn. Prentice Hall, Upper Saddle River, p 566
2. Boehm BW, Abts C, Brown AW, Clark BK, Horowitz E, Madachy R, Reifer D, Steece B (2000) Software cost estimation with COCOMO II. Prentice Hall, Upper Saddle River
3. Cabrera D, Cabrera L (2015) In: Powers E (ed) Systems thinking made simple: new hope for solving wicked problems. Odyssean Press, p 59

4. Carter A (2010) Better buying power: mandate for restoring affordability and productivity in defense spending [memorandum]. Department of Defense, Washington, DC
5. Carter A (2015) Message from secretary ashton carter to all department of defense personnel. Department of Defense
6. Department of defense instruction (2015) Operation of the defense acquisition system (Instruction No. DoDi 5000.02). Under Secretary of Defense for Acquisition, Technology, and Logistics (USD(AT&L)), Washington, DC
7. Farr J, Valerdi R, Goerger S, Dabkowski M (2016) Conversation between subject matter experts in field of cost estimation on paradigms of cost for generic systems. West Point
8. Ford TC (2008) Interoperability measurement. Air Force Institute of Technology Wright-Patterson Air Force Base, Ohio. Graduate School of Engineering and Management
9. GAO (2014) F-35 lightning II program (F-35) (Assessments of Major Weapons Programs No. 14-340SP). Washington, DC: US Government Accountability Office
10. Goerger SR, Madni AM, Eslinger OJ (2014) Engineered resilient systems: a DoD perspective. Conference on Systems Engineering Research
11. Holland JP (2016) Engineering for resilience. SciTech Presentation, San Diego. Slide 7
12. Under Secretary of Defense (AT&L) (2015) Operation of the defense acquisition system (No. DoDI 5000.02). Under Secretary of Defense (AT&L)
13. SEER-H® Documentation Team: MC, WL, JT, KM (2014) SEER for hardware detailed reference – user's manual. United States of America: Galorath Incorporated (User's Manual)
14. Shibata JT (2010) A space acquisition leading indicator based on system interoperability maturity, No. AFIT/GSE/ENV/10-D02DL. Air Force Institute of Technology Wright-Patterson Air Force Base, Ohio. Graduate School of Engineering and Management
15. Valerdi R (2008) In: Muller D (ed) The constructive systems engineering cost model (COSYSMO): quantifying the costs of systems engineering effort in complex systems. VDM Verlag, Saarbrucken
16. Valerdi R, Ryan TR (2016) Total cost of ownership (TOC): an approach for estimating UMAS costs. In: Caras JR, Dickmann JQ (eds) Operations research for unmanned systems, 1st edn. Wiley, Hoboken, pp 207–232
17. Young L, Wade J, Valerdi R, Farr J, Kwak YH (2010) An approach to estimate the lifecycle cost and effort in project management for systems centric projects. International Forumn on COCOMO and Systems/Software Cost Modeling
18. Wang G, Valerdi R, Roedler GJ, Ankrum A, Gaffney JE (2012) Harmonising software engineering and systems engineering cost estimation. Int J Comput Integr Manuf 25 (4–5):432–443. doi:10.1080/0951192X.2010.542182

Chapter 3
Introducing Resilience into Multi-UAV System-of-Systems Network

Edwin Ordoukhanian and Azad M. Madni

Abstract Unmanned aerial vehicles (UAVs) are used in a variety of missions such as surveillance, law enforcement, agriculture, search and rescue, and payload delivery. Multiple UAVs are simultaneously deployed by both government and civilian organizations to achieve superior coverage and a more flexible response to disrupting events. In this chapter, multi-UAV systems are discussed from a system-of-system (SoS) perspective. It is hypothesized that the key advantage of this approach is the flexibility afforded to study different interaction protocols and conduct trade-offs in terms of both resource allocation and function allocation to the different members in the multi-UAV SoS. It is also hypothesized that this approach will allow studying of SoS resilience and resilience approaches, and if resilience approaches are chosen systematically the chances of safe and fast recovery will increase significantly while dealing with disruptions. This chapter also discusses single- and multi-UAV modeling and presents preliminary simulation results.

Keywords Multi-UAV operation • Quadcopter • Resilient systems • Resilience approach

3.1 Introduction

Today, there is a high demand for unmanned aerial vehicles (UAVs). These vehicles have many application domains such as military reconnaissance, surveillance, scientific data collection, search and rescue, and payload delivery. Designing and operating a single vehicle that meets all mission requirements is costly, labor-intensive, and requires a longer schedule [1]. Furthermore, single vehicle operation can experience down time due to internal failures, scheduled maintenance, or refueling. These, in fact, disrupt the system operation and put mission success at risk [2].

E. Ordoukhanian (✉) • A.M. Madni
University of Southern California, Los Angeles, CA, USA
e-mail: Ordoukha@usc.edu

On the other hand, multi-UAV operation enables allocation of mission requirements to different vehicles. This further enables flexibility in allocating requirements to multiple vehicles, which reduces design complexity [2]. It enables flexibility in allocating functionalities to different vehicles, which increases overall mission coverage. Multi-UAV operation has multiple simultaneous interventions. Component systems collect information from multiple sources (through their sensors) and apply forces (actions) simultaneously at different locations [3]. Operating multiple vehicles simultaneously also has greater time efficiency as missions can be completed faster by each vehicle performing a specific task toward overall mission objective [3]. Furthermore, due to heterogeneity of multi-UAV operation (i.e., multiple vehicles having different capabilities), the capabilities of vehicles complement each other, which contributes toward system resilience. In addition, multi-UAV operation has higher reliability since each vehicle has some degree of fault tolerance. Multi-UAV operation also enables flexibility and adaptability in terms of communication protocols and functional allocations. These contribute to system resilience [3].

This chapter is organized as follows. Section 3.2 discusses the multi-UAV complex as a system-of-system (SoS). Section 3.3 discusses resilient multi-UAV operation and presents the key resilience definitions and concepts in multi-UAV context. Section 3.4 discusses current gaps and challenges in introducing resilience into multi-UAV systems. Section 3.5 discusses current research and research hypothesis. Section 3.6 discusses single-UAV and multi-UAV modeling approach and presents simulation results. Section 3.7 concludes the accomplishments to date and discusses the way ahead.

3.2 Multi-UAV as a System-of-System

*In multi-UAV system-of-system, e*ach vehicle executes a set of tasks (goal and subgoals) to achieve the global mission. Task allocation, high-level planning, plan decomposition, and conflict resolution should be solved taking into account global mission and UAV capabilities. Coordination and Cooperation are other important aspects that need to be taken into account. Multi-UAV coordination deals with sharing resources, both temporal and spatial. Temporal coordination relies on synchronization among vehicles with respect to their operation. Spatial coordination deals with sharing of the space among vehicles to ensure safe operation. This is important to avoid colliding into each other while avoiding dynamic and static obstacles. In this case, planning algorithms play a key role. In practice, sampling-based algorithms are most likely to scale well. Multi-UAV cooperation is joint collaborative behavior that is directed toward some goal in which there is common interest or reward. Cooperation of heterogeneous vehicles requires integration of sensing, control, and planning in an appropriate decision architecture [3].

Centralized and decentralized decision-driven architecture can be employed. However, the decision about the choice of the architecture depends on each vehicle's sensing capability, computational power, and system scalability. In centralized decision-driven architectures, computational capabilities should be compatible with the amount of information to process. In this architecture, exchange of data should meet both speed requirements (keep data up-to-date) and expressivity (quality of information enabling well-informed decision taking). On the other hand, in distributed decision-driven architectures, the available knowledge within each distributed vehicle is sufficient to make "coherent" decisions. This architecture has superior suitability to deal with system scalability [3].

In multi-UAV system-of-system, each UAV has some degree of autonomy and intelligence. The control of Multi-UAV SoS is focused on trade-off analysis on system and SoS levels, which is a prerequisite to study resilience [4].

In a multi-UAV SoS, vehicles have *operational independence* as each system operates to perform its assigned function while also participating in the SoS put together to carry out the overall mission [4–6]. The vehicle can also have different *governance* while participating in the SoS. Multi-UAV SoS *evolves* with functions and purposes added, removed, and modified with experience and with changing needs or mission objectives [4–6]. Multi-UAV SoS exhibits *emergent* behavior as SoS overall functionality does not reside within any single UAV (i.e., multi-UAV SoS behavior cannot be realized by a single UAV). UAVs are *geographically distributed* since they primarily exchange information – not mass or energy [4, 5, 7].

Multi-UAV SoS also exhibits complex adaptive system's characteristics such as *heterogeneity*, the degree of diversity in interactions that can range from uniform (no diversity) to unique (extreme diversity); *randomness*, degree of patterns or predictability in interactions that ranges from highly predictable to unpredictable; and *modularity*, the degree of clustering or grouping in interactions that ranges from no-clustering to tight clustering [4].

3.3 Multi-UAV Resilient Operation

Multi-UAVs, operating in open environments, are susceptible to disruptions. Uncertainties and unexpected factors in the environment disrupt the system's operation and adversely impact overall system performance. Thus, the system should be able to deal with these disruptions and operate within the dynamic environment while maintaining acceptable levels of performance [2, 4, 8]. The ability to maintain acceptable levels of performance through flexibility and adaptability in the face of disruptions is called *resilience* [8].

Disruptions can be categorized into three main groups. External disruptions are associated with environmental obstacles and incidents [8]. They are often random and with unknown severity and duration [8]. Systemic disruptions happen when an internal component's functionality, capability, or capacity causes performance degradation. They are easily detectable in technological systems [8]. Human-triggered disruptions are associated with human operators inside or outside of the system boundary impacting system performance. In general, these disruptions can be predictable or random [8].

There are many definitions of resilience. Not all apply to multi-UAV operation. The applicable definitions are to maintain acceptable level of service (performance) in the face of interruptions during system's normal operation [8–10]; to serve effectively in a variety of missions with multiple alternatives through rapid reconfiguration or timely replacement despite uncertainties about individual system performance [8–10]; and to anticipate, resist, absorb, respond to, adapt to, and recover from both natural and man-made disruptions [8–10]. One key resilience heuristic in designing multi-UAV SoS is *Loose Coupling,* which means that component systems must be loosely coupled to assure ease of change in interactions among component systems [8, 11].

Resilient multi-UAV SoS have the following characteristics: (a) *localized capacity*, if a UAV is damaged, remaining UAVs should be able to take over the functions of the incapacitated UAV; (b) *collision avoidance*, UAVs are able to detect and avoid obstacles; and (c) *reconfiguration and maneuverability* of the entire system-of-system (i.e., presence of some obstacles may not be known due to uncertainties in operational environment); and (d) *extending capacity and capability* to deal with disruption (e.g., manually with human intervention or autonomously) [8, 12, 13].

There are metrics to monitor performance of the resilient multi-UAV SoS. These include metrics such as *flow rate* that is the speed at which data are being transferred between vehicles and to the ground station. Accurate and timely data flow are essential to successful operation. This also heavily depends on effective, reliable, and secure communication between UAVs and ground station. The overall state awareness of SoS depends on data flow. *Response time* is the round-trip time between sending a command to the multi-UAV SoS and receiving a response. Factors such as task distribution algorithm, individual system's capabilities, and communication bandwidth and protocols have an impact on response time. *Recovery time* is the time it takes between detecting a failure and restoring full or partial operation [2, 4].

Figure 3.1 presents a conceptual schema for resilient multi-UAV system-of-systems. Multi-UAV SoS is required to satisfy mission objectives. The mission objective can be broken down into three levels: SoS level, system (i.e., UAV) level, and the constraints imposed by the operational environment. While satisfying mission objectives, a multi-UAV SoS will invariably have to respond to disruptions. A resilient multi-UAV SoS employs resilience approach (RA) to deal with disruptions. Resilience approaches explicitly take into account the type of disruptions that the system is likely to encounter, system-of-system type (which is dictated by mission objectives), and multi-UAV coordination and cooperation. The latter is constrained by SoS type [2, 4].

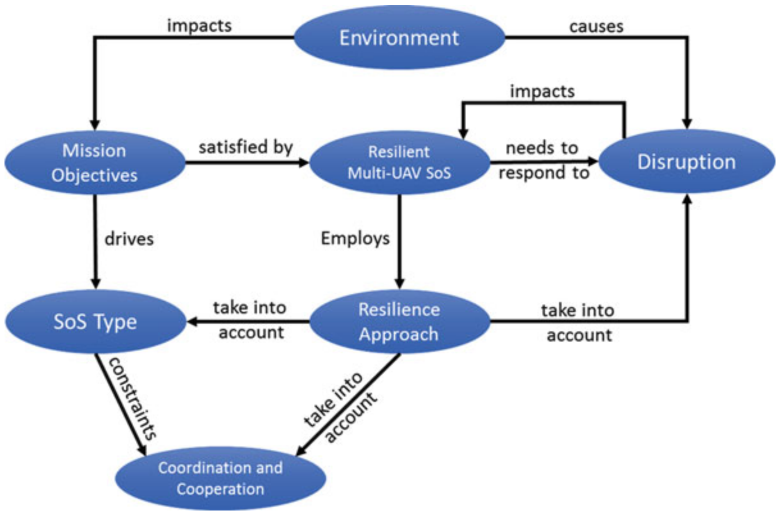


Fig. 3.1 Resilient multi-UAV SoS conceptual schema

3.4 Current Status and Challenges in Introducing Resilience into Multi-UAV SoS

Current resilient methods and approaches are ad hoc and are mostly tailored toward a specific mission and operational context. They often tend to be complex and costly as they require anticipating potential disruptions in order to have a plan ready to deal with and recover from the disruption. Current techniques allow early identification of levers (i.e., potential disruptions) in the operational context to identifying key areas of resilience [2, 4]. Trade-off and risk analyses tied to mission context and underlying physics play important roles in resilient system design. Specifically, designing a resilient multi-UAV system requires in-depth trade-off studies that span both SoS and individual system levels [2, 4]. State estimation is another resilient method that enables determination of the current state and path to desired end state, or neutral state. During resilient system design, trade-space exploration allows designers to uncover hidden interactions among UAVs, and to understand how change propagates. This leads to incorporating measures such as adding resources, adding margins, and increasing capacity to counteract [2, 4].

Designing a resilient multi-UAV system poses several challenges. Designing for resilience significantly impacts both cost and development schedule. It is not cost-effective to design a multi-UAV complex that deals with infrequent, inconsequential disruptions; thus, disruptions need to be prioritized, and the most consequential disruptions must have priority. However, the system should still be able to deal with infrequent, inconsequential disruptions. During design time, likelihood of conflicts among requirements increases with an increase in the number of UAVs in the SoS. Therefore, an organizing framework is needed to manage system requirements. As system complexity increases, it becomes more vulnerable and less reliable; thus, system elegance plays a vital role (i.e., minimizing structural complexity). Heterogeneity of UAVs impacts resilience approach design and implementation. To deal with a particular disruption or set of disruptions, multiple resilience approaches can be employed. However, not all approaches will be affordable. Additionally, it is necessary to ensure compatibility of resilience approaches with system and SoS capabilities. To perform trade-offs among multiple resilience approaches, we have to define a metric to measure the effectiveness of a resilience approach. Additionally, these metrics need to be evaluated through simulation. Verification and validation of a resilient multi-UAV system is time-consuming and costly as system behavior is not fully predictable. Due to nondeterminism, situational (state) awareness in multi-UAV is a challenging task. Therefore, we need to employ probabilistic models [5, 14, 15].

3.5 Existing Gap and Current Research

What is currently missing in resilient system and system-of-system (SoS) design is a methodology that allows systematic identification of resilience approaches and then conducts comparative trade-off analysis among them for both system and system-of-system (SoS) levels.

It is hypothesized that by framing the multi-UAV problem as a SoS problem the key advantage is the flexibility afforded to study different interaction protocols and conduct trade-offs in terms of both resource allocation and function allocation to the different members in the SoS. Furthermore, it is hypothesized that these capabilities will enable the investigation of resilience and resilience approaches within SoS. It is also hypothesized that if resilience approaches are chosen systematically, the chances of safe and fast recovery will increase significantly when dealing with disruptions.

Resilience approach (RA) is a series of actions or steps taken to perform one or more of the following: anticipate, resist, absorb, respond to, adapt to, and recover from disruptions. An RA can be an algorithm or a set of rules to deal with known and unknown disruptions. The actions taken by RA can be categorized into two groups: actions such as anticipation and disruption detection can be categorized as "observation"; and actions such as adapting and responding to a certain disruption can be categorized as "Guidance and Control." Therefore, in this context, resilience is the ability to perform two key functions: observation and detection, and guidance and control. However, to perform these two functions, other activities such as data analytics, context management, and real-time trade-off analysis should also be performed. Example resilience approaches are circumvention, recovery, and reconfiguration [2, 4].

RA also takes into account system-of-system (SoS) type, disruption type and severity, operational context, and multi-UAV coordination and cooperation. Disruption severity is a function of operational context and the duration of disruption.

Table 3.1 Factors influencing resilience approach

	Factors
SoS level	Configuration (e.g., virtual, collaborative, acknowledged, and directed)
	Communication protocols and mechanism (e.g., direct communication among vehicles, via satellite)
	Disruption type and severity (e.g., communication loss, extreme wind gust)
	Coordination and cooperation
	Task distribution algorithm
System level	System architecture (e.g., modular, integrated)
	Physical characteristics (e.g., shape, size, and weight)
	Hardware/software capabilities
	Disruption type and severity (e.g., communication device failure)

The context reflects the system's current state and current operational use. The duration of disruption determines whether a disruptive event is temporary or an indication of a trend. Other factors on system and SoS levels impact the identification of the right resilience approach. These factors are summarized in Table 3.1 [2, 4].

Dealing with disruption consists of multiple steps. First, disruption occurrence needs to be detected, and disruption information needs to be propagated through SoS. Depending on the disruption type and severity, an appropriate action must be taken by the individual systems or SoS. These actions must be coordinated in parallel. However, the decision to move forward to execution depends on improving the SoS-level performance condition. This requires multiple decision-making loops and performing real-time trade-off analysis among multiple alternatives. Once the final decision has been made to move forward, the system executes the decision and verifies whether or not it satisfies the mission objective. If objective is not satisfied, then the system requires an adjustment to its course of action. Figure 3.2 shows a notional flow of dealing with disruptions [2, 4].

For example, if there is a communication loss, an appropriate and suitable communication path with sufficient resources (i.e., bandwidth) has to be found to replace the current communication path. Being a suitable choice is a function of context. In general, the pre-established backup communication path has to be separate from the current communication path. Resources and functions need to be allocated to the backup path. Finally, the communication has to be switched over to the backup path. The described actions may take place at different points in time.

To deal with disruptions, multiple resilience approaches can be employed. Table 3.2 represents a list of resilience approaches and maps them into two categories: system level, and system-of-system level. Some of these approaches are easier done on the SoS level than on the system level. Some of the approaches can be done on both levels. Employing a resilience approach on either level (SoS/system) has implications on the other level. Therefore, to choose an appropriate resilience approach, these implications must be taken into account and necessary trade-offs must be considered [2, 4].

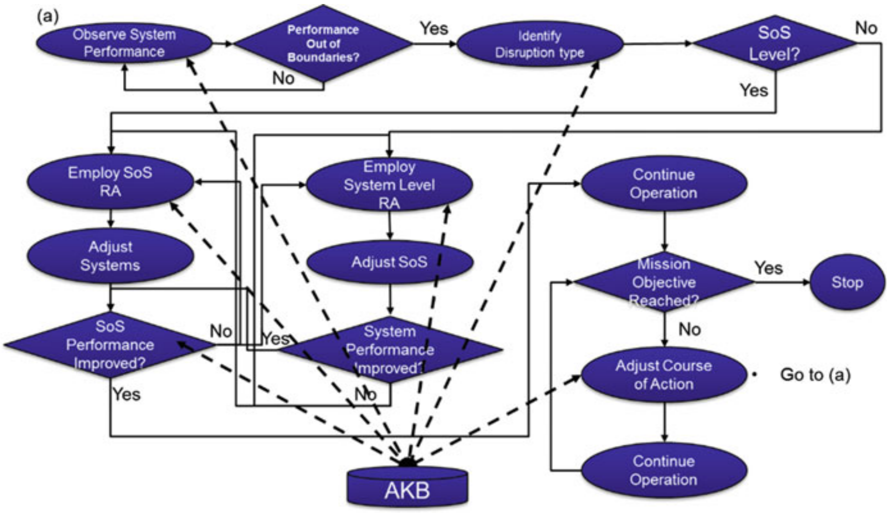


Fig. 3.2 Notional flow for dealing with disruptions

Table 3.2 Resilience approach classification

Resilience approach	SoS	System	Invoking disruption
Functional redundancy	■	■	Systemic, external
Functional reallocation	■		Systemic, external
Physical redundancy	■	■	Systemic, external
Reorganization	■		External
Reconfigurability	■		Systemic, external
Recovery	■	■	Systemic, external, human-triggered
Circumvention	■	■	Systemic, external,
Human-in-the loop (human as a backup)	■	■	Systemic, external, human-triggered
Anticipation	■		External
Drift correction	■	■	Systemic, external
Graceful degradation	■		External
Neutral state (preplanned protocol)	■	■	Systemic, external, human-triggered
Adaptation	■		External

Some of the resilience approaches (e.g., *functional redundancy*) can be viewed as rules [8]. These rules can be taken into account at design time, as well as during system operation. To actually follow these rules during system operation, the system would require a procedure, or an algorithm. For example, to accomplish the same functionality by other means, the system would need to search through “in-house” capabilities and find the combination of component systems that would be able to accomplish that functionality. A practical example would be the following. Assume that there is a UAV within the multi-UAV SoS that is capable of taking high-resolution pictures and sending them to the ground station through a high-bandwidth communication link. If, due to an external disruption, that UAV is no longer able to perform its function, the system would switch that function over to two separate UAVs, one capable of a high-bandwidth communication link and the other capable of taking high-resolution pictures. However, to do that the system should "search" through the SoS and identify specific components to satisfy the primary functionality. This process can be viewed as an "algorithm." [2, 4, 8].

Resilience approaches can also take the form of algorithms. For example, anticipation can be implemented as an algorithm with the system or system-of-system constantly predicting what would happen next. To do that, the system/SoS should have a good estimate of the current state and an "understanding" of the consequences of its action to be able to predict the next move. To successfully do that, it would require an input from a "monitoring" algorithm.

Resilience approaches mentioned in Table 3.2 are briefly discussed next. *Functional redundancy* is achieving the same functionality with other means [8]. *Function reallocation* means to redistribute tasks among the remaining component system upon the loss of a system (disruption). *Physical redundancy* is to have another system take over when one system fails, or have another subsystem in a system to take over when one subsystem fails. *Circumvention* is avoiding or bypassing a disruption. *Reorganization* means the system should be able to restructure itself when dealing with a disruption. *Reconfigurability* is when system's behavior changes in order to deal with a disruption. *Recovery* is system's ability to go back to the last known state before the disruption or a better state. *Human as a Backup* means if a component system fails, human is brought into the loop to deal with the disruption. *Anticipation* is constant prediction of what would happen to the system or system-of-system. *Drift correction* is to initiate counter measures before onset of disruptions. *Graceful degradation* means system's performance degrades gracefully outside the performance envelope. *Preplanned protocols* mean that if communication between systems fails, go to a neutral state or initiate a predefined procedure. *Adaptation* means dealing with unknown disruptions and adjusting system behavior in the process [8].

3.6 Modeling Multi-UAV SoS and Simulation

Quadcopters are widely being used for research purposes. They are easier to model than fixed-wing RC planes and are suitable to demonstrate resilient behavior in SoS model. The quadcopter model implemented in this research effort has adequate fidelity to answers key questions based in this research. It captures nonlinear dynamics of quadcopters, and it takes into account aerodynamic drag coefficients for translational motion. The model is able to go through waypoints, or follow a specified trajectory. It has a basic sensor model. It is flexible enough to introduce and test-drive resilience concepts.

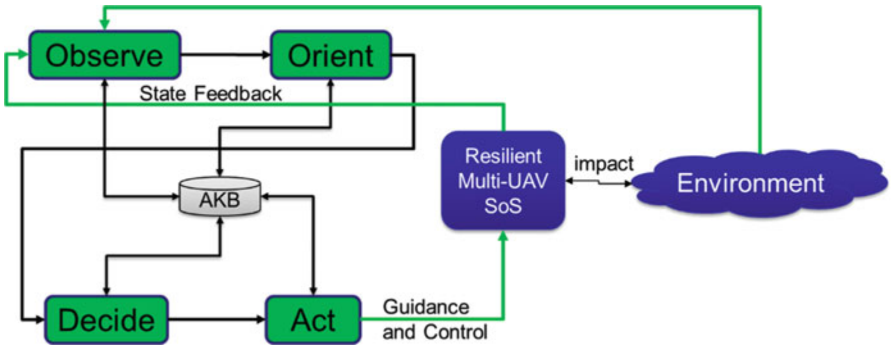


Fig. 3.3 Overall modeling construct for multi-UAV SoS

Figure 3.3 shows overall modeling construct for multi-UAV SoS. This is similar to the Observe, Orient, Decide, Act (OODA) loop introduced by John Boyd [16]. However, it has some extra elements such as recording observations, data analysis, decision, and actions into an active knowledge base (AKB). Active knowledge base dynamically stores and updates a set of facts, assumptions, and rules that the system uses to deal with disrupting events.

Figure 3.4 shows overall architecture of the quadcopter. The vehicle is equipped with sensors and a communication device. Sensors are used to collect internal information about the vehicle, and external information about the environment. This information is then shared with neighboring vehicles within the SoS through the communication interface. This device also receives information from neighboring vehicles. The vehicle also has state estimator. The state estimator estimates system's current state taking inputs from sensors and communication device. It sends signal to "Event Triger" module. Event trigger module determines if the combination of system's current state, sensors data, and messages from the communication exceeds the acceptance threshold, and whether a system response is required. Then this information is passed into "Decisional and Control Law Module" where proper decision is made and a corresponding control signal is sent to the vehicle. Active knowledge base "AKB" is responsible for capturing system's current state, triggering events, system's decision, and operational context.

A smaller fragment of Figs. 3.3 and 3.4 for multi-UAV operation with three quadcopters is implemented in MATLAB/Simulink. The implemented quadcopter model is shown in Fig. 3.5. The simulation result is shown in Fig. 3.6. In Fig. 3.5, "Quadcopter Nonlinear Dynamics" captures quadcopter's nonlinear equations of motion with aerodynamic drag coefficients. Attitude and position controllers are nonlinear controllers responsible for holding the desired position and attitude of quadcopters. The sensors are on a feedback path and are feeding back the signals to the controllers with some error. "Autopilot" block is responsible for guiding the quadcopter in the environment by avoiding static obstacles.

The "Autopilot" block passes the desired trajectory or waypoint signals directly to the position controller during normal operation. "Obstacle Detection" block

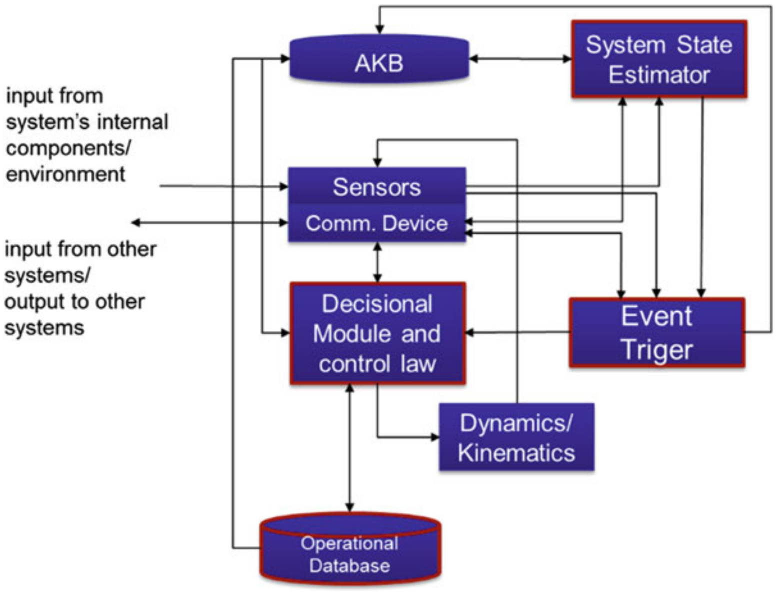

AKB
System State Estimator
input from system's internal components/ environment
Sensors
Comm. Device
input from other systems/ output to other systems
Decisional Module and control law
Event Triger
Dynamics/ Kinematics
Operational Database

Fig. 3.4 Vehicle's overall architecture

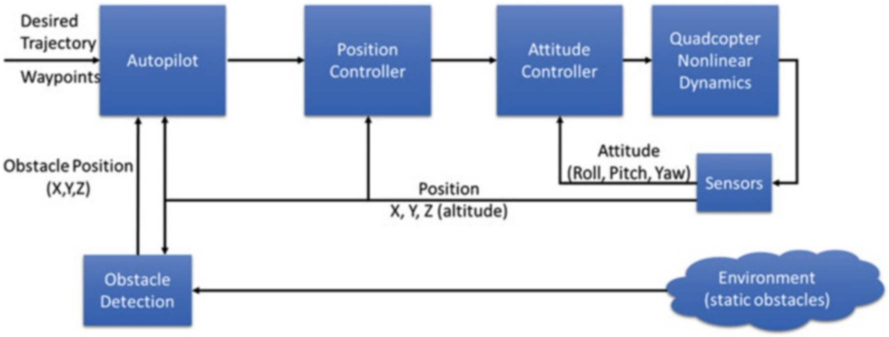

Desired Trajectory Waypoints
Autopilot
Position Controller
Attitude Controller
Quadcopter Nonlinear Dynamics
Obstacle Position (X,Y,Z)
Attitude (Roll, Pitch, Yaw)
Sensors
Position X, Y, Z (altitude)
Obstacle Detection
Environment (static obstacles)

Fig. 3.5 Implemented model

constantly monitors and determines if the distance between the quadcopter and static obstacle is less than a safe operating distance. If it is, then it sends obstacle's coordinates to "Autopilot" block. "Autopilot" block then stops sending desired trajectory signals to the quadcopter and sends a new set of coordinates to the vehicle in order to avoid the obstacle.

The simulation setup is following. The quadcopters start from (0,0,0) position. They each have to follow the desired path (shown in dotted line). There are static obstacles distributed in the environment and quadcopters should avoid these obstacles from either left or right, if the distance between the quadcopter and obstacle is

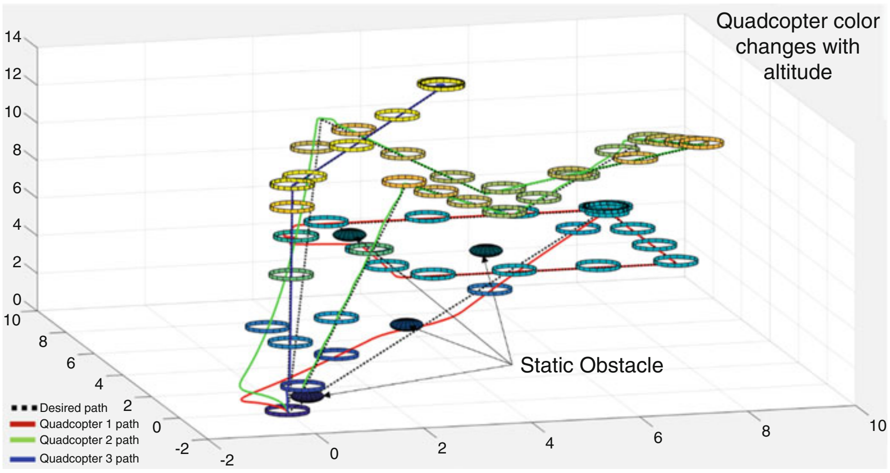


Fig. 3.6 Quadcopters desired and followed paths

less than 1 m. In this scenario, since quadcopters are all going in a different direction the chances of quadcopters bumping into each other are low.

Quadcopters 2 and 3 follow their assigned paths without any problem since there are no obstacles. Quadcopter 1 has to make some maneuvers and diverge from its assigned path since there are obstacles on its way. The vehicle avoids the first obstacle from left and the second obstacle from right-bottom. When the distance between a quadcopter and an obstacle is less than 1 m, the "Autopilot" stops sending the desired trajectory information to the quadcopter and begins sending new x,y,z coordinates to the vehicle. Due to vehicle dynamics and controller response time, it takes a finite amount of time for the quadcopter to respond to new coordinates.

Figure 3.6 shows a multi-quadcopter SoS demonstrating basic levels of resilience. The vehicles are able to follow an assigned path and avoid obstacles. The "Autopilot" and "Obstacle Detection" blocks enable following resilience approaches on the system level mentioned in Table 3.2: Drift correction, circumvention, and recovery. Drift correction occurs when the "Obstacle Detection" block detects that the distance between the current position and obstacle is less than the safe operating distance. It then sends a signal to the "Autopilot" block to initiate circumvention, which allows the vehicle to avoid the obstacles. Once it is determined that there are no obstacles and "Obstacle Detection" block determines that it is safe to return to the original path, it sends a signal to the "Autopilot" to initiate the recovery process and steer the vehicle back to the desired path.

3.7 Summary and Way Ahead

Unmanned aerial vehicles (UAVs) are used in a variety of missions such as surveillance, law enforcement, agriculture, search and rescue, and payload delivery. Operating multiple UAVs simultaneously allows superior coverage and greater flexibility when responding to disrupting events. Multi-UAV systems can be viewed from a different perspective. This chapter looked at multi-UAV complex as a system-of-system (SoS). It is hypothesized that the key advantage of this approach is the flexibility afforded to study different interaction protocols and conduct trade-offs in terms of both resource allocation and function allocation to the different members in the multi-UAV SoS. It is also hypothesized that these capabilities will enable the investigation of resilience approaches within SoS. It is also hypothesized that if resilience approaches are chosen systematically, the likelihood of safe and fast recovery will increase significantly when dealing with disruptions.

In this chapter, a quadcopter model is employed and simulation results for a basic level of resilience are shown. Future steps involve expanding the model to include different kinds of disruptions as well as multi-UAV coordination and cooperation.

References

1. Ordoukhanian E, Madni AM (2015) System trade-offs in multi-UAV networks. AIAA SPACE 2015 conference and exposition; 2015/08/28: American Institute of Aeronautics and Astronautics (AIAA)
2. Ordoukhanian E, Madni AM (2016) Resilient multi-UAV operation: key concepts and challenges. 54th AIAA Aerospace Sciences Meeting; 2016/01/02: American Institute of Aeronautics and Astronautics (AIAA)
3. Valavanis KP, Vachtsevanos GJ (2014) Handbook of unmanned aerial vehicles. Springer Publishing Company, Incorporated
4. Ordoukhanian E, Madni AM (2016) Toward development of resilient multi-UAV system-of-systems. AIAA SPACE 2016:5414
5. Madni AM, Sievers M, Humann J, Ordoukhanian E, Boehm BW, Lucero S (eds) Formal methods in resilient systems design: application to multi-UAV system-of-systems control. Conference on Systems Engineering Research (CSER) 2017, Redondo Beach Springer
6. Maier MW (ed) (1996) Architecting principles for systems-of-systems. INCOSE international symposium, Wiley Online Library
7. Maier MW (1996) Architecting principles for systems-of-systems. INCOSE Int Symp 6 (1):565–573
8. Madni AM, Jackson S (2009) Towards a conceptual framework for resilience engineering. IEEE Syst J 3(2):181–191
9. Neches R, Madni AM (2013) Towards affordably adaptable and effective systems. Syst Eng 16 (2):224–234
10. Goerger SR, Madni AM, Eslinger OJ (2014) Engineered resilient systems: a DoD perspective. Procedia Comput Sci 28:865–872

11. Humann J, Khani N, Jin Y (2014) Evolutionary computational synthesis of self-organizing systems. Artif Intell Eng Des Anal Manuf 28(3):259–275
12. Uday P, Marais KB (2014) Resilience-based system importance measures for system-of-systems. Procedia Comput Sci 28:257–264
13. Uday P, Marais K (2013) Exploiting stand-in redundancy to improve resilience in a system-of-systems (SoS). Procedia Comput Sci 16:532–541
14. Madni AM, Sievers M (eds) (2015) A flexible contract-based design framework for exaluating system resilience approaches and mechanisms. IIE annual conferenec and expo, 30 May–2 June, Nashville
15. Sievers M, Madni AM (eds) (2014) A flexible contracts approach to system resiliency. Systems, Man and Cybernetics (SMC), 2014 I.E. International Conference on: IEEE
16. Boyd JR (1996) The essence of winning and losing. Unpublished lecture notes

Chapter 4
Considerations for Engineered Resilience from Examples of Resilient Systems

Rosalind Lewis

Abstract Everyday commercial, civil, and defense enterprises are faced with disruptive events that have the potential to degrade or altogether impede business as usual. In an increasingly interconnected and interdependent world, where systems/system-of-systems provide functionality enabling operational capabilities, complexity is commonplace; and the ability to anticipate and therefore manage all potential disruptions becomes untenable. Recognizing this dilemma, organizations are trying to understand and infuse the properties of resiliency in their culture, processes, and assets. Resilience is a widely used term and in general is understood as the ability to operate through some adverse condition. Accordingly, engineered resilience, the notion of designing resilience into a system from the outset, is a frequent subject of analysis and research in the systems engineering and acquisition community. This paper explores the question of what it means to purposefully engineer resilient systems by examining systems that displayed resilience, thus continuing to provide a capability even through disruptions. The examples – an acquisition system and an operational system – are analyzed with respect to various resiliency concepts. This includes applying several resiliency definitions to the example systems and identifying metrics for measuring resilience which introduces the notion of drift, timeliness, and process as important resiliency factors. Finally, observations related to engineering resilient systems are offered.

Keywords Engineered resilience • Resiliency • Disruption • Systems • Process

4.1 Introduction

Engineering resilient systems is an important business matter. Everyday commercial, civil, and defense enterprises are faced with disruptive events that have the potential to degrade or altogether impede business as usual. In an increasingly interconnected and interdependent world, where systems/systems-of-systems

R. Lewis (✉)
University of Southern California, Systems Architecting and Engineering Program, Los Angeles, CA, USA
e-mail: lewisr@usc.edu

A.M. Madni et al. (eds.), *Disciplinary Convergence in Systems Engineering Research*, https://doi.org/10.1007/978-3-319-62217-0_4

provide functionality enabling operational capabilities, complexity is commonplace; and the ability to anticipate and therefore manage all potential disruptions (changes, events, or conditions that adversely impact the system performance) becomes untenable. Complexity has the capacity to enable, as well as mask emergent behaviors, increasing the potential for unexpected or unpredictable conditions. Recognizing this dilemma, organizations are trying to understand and infuse the property of resiliency in their culture, processes, and assets.

Accordingly, engineered resilience, the notion of designing resilience into a system from the outset, is a frequent subject of analysis and research in the systems engineering and acquisition community. Resilience is a widely but inconsistently used term, with many formal definitions equating it to a property, capability, and/or process. This paper explores the question of what it means to purposefully engineer resilient systems by examining systems that displayed resiliency, thus continuing to provide a capability even through disruptions. The two examples are (1) an acquisition system (AcqSys) which exists primarily to create a product and (2) an operational system (OpSys) which exists primarily to provide a service. These systems are analyzed with respect to various resiliency concepts including the characterization using several resiliency definitions and the introduction of resiliency metrics that suggest the notion of drift, timeliness, and process as important resiliency factors. Combining these with the growing body of literature and research, observations related to engineering resilient systems are offered.

The paper is divided into six sections. Section 4.2 describes the AcqSys and OpSys examples used to consider resilience and the engineering of resilient systems. Section 4.3 describes the characteristics of the example systems that drive the need for resilience. Section 4.4 evaluates how these systems handled a small set of disruptions, to explore the exhibited behaviors as compared to resiliency definitions. Section 4.5 considers resilience as a process (which can be measured) and is subsequently used to construct a concept of operations (CONOPs) for resiliency. Section 4.6 concludes by offering observations on purposefully engineering resilient systems.

4.2 An Acquisition and an Operational System – As Examples

In order to consider resiliency in context, it is helpful to have demonstrable systems for consideration; the selection of which itself might be highly debatable. For example, what would be a suitable exemplar for analysis: a nonresilient system, a somewhat resilient system, or a highly resilient system? What distinguishes them one from another? Rather than attempt to resolve that debate, systems that spanned different parts of the life cycle and demonstrated resilience were selected without consideration for some defined level of resilience. The question was simply – what was the response that enabled these systems to continue through disruptions? More specifically what can we learn about resiliency by examining the “response” to disruption? A starting point to understanding the “response” can be found in understanding the “effort.”

A system “a construct or collection of different elements that together produce results not obtainable by the elements alone” [16] expends some *effort* to provide a *capability*. This effort can be measured by the resources required to provide a capability both temporally and in magnitude (e.g., money, energy, staff, etc.). When disruptions occur, the anticipated performance of a system (some relationship between effort and capability) may decline or stop altogether. A system that demonstrates resiliency may need to reconfigure, operate in a degraded mode, alter timelines, or some combination thereof; in order to continue providing useful capability despite the disruption. This activity can be considered as the effort-to-continue-through-disruption ($effort_{td}$) to provide a capability-through-disruption ($capability_{td}$), which can also be measured by the resources required.

AcqSys or OpSys routinely encounters problems (unavoidable conditions, unmitigated consequences, or unforeseen circumstances) that are disruptive and capable of modifying the effort required to deliver a product or service. For instance, the creation of a new aircraft may include functions such as design, contracting, research, manufacturing, verification, and a host of many systems engineering processes. The *effort* duration to create a new aircraft spans many life cycle phases [6] such as exploratory, concept, development, and production, whereas the *effort* magnitude will rise and fall in accordance with the scope of work to be accomplished. For the purposes of this paper, the Boeing 787 acquisition system (787AcqSys) is used as an example of the *effort to create a product* (the Dreamliner). This system experienced several problems during execution. When the 787AcqSys did not execute in accordance with the anticipated performance (i.e., cost, schedule, or technical targets) due to disruptive events, the customer needs and/or the supplier objectives were at risk until some $effort_{td}$ was implemented, which in this case increased the overall effort (both in time and money) required for development [see Appendix A].

Similarly, consider systems that provide a service, such as mass transportation systems, which are frequently composed of many distinct transit systems owned and operated by a variety of agencies and organizations. The collection of systems inherently provides flexible and adaptable service. The *effort* duration for providing a mass transit service spans the utilization and support phases of the life cycle, and the *effort* magnitude is a function of stakeholder’s needs and available resources to leverage/operate mass transit assets. For the purpose of this paper, a mass transit operational system (MTOpSys), like those employed by the cities of Los Angeles and Washington DC, is used an example of the *effort to provide a service*. Such systems frequently experience problems during operation. When MTOpSys does not perform in accordance with expectations, due to disruptive events, the service a user experiences is subject to alteration or degradation while some $effort_{td}$ to provide a $capability_{td}$ is expended or suspension until some effort to restore ($effort_r$) the *capability* is accomplished. When tunnel fires disrupted service lines in the Washington DC metro, users had to take alternate routes ($capability_{td}$) until the service lines could be repaired. The metro system spent resources (*effort*$_{td}$ and *effort*$_r$) to continue service through disruption and achieve restoration.

4.3 The Need for Resilience

As suggested above handling disruptive events is an important business matter; and the pursuit of resilience is in part an acknowledgment that predicting and mitigating all potential disruptions are neither practical nor feasible, particularly in complex environments. Complexity is an enabler and concealer of emergence where unexpected and unpredicted occurrences emanate. It is correct to characterize the 787AcqSys and MTOpSys as complicated systems, but it is also correct to characterize them as complex systems.

Complicated systems are systems where the behavior "of the whole can be *constructed* from behavior of parts" [10]. The objective or desired behavior is what is expected, based on system composition. The design is an arrangement of elements that collectively provides a requisite functionality for the objective behavior of the whole. Each component of the Dreamliner (see Fig. 4.5) was the product of a different 787AcqSys element. The 787AcqSys could not have fulfilled its purpose without those elements. The components of the MTOpSys enable a service (broad area mass transit) that would not be attainable by any component alone. Each part has a role in the greater purpose, which collectively was designed to achieve an objective behavior, as depicted by the system architecture.

However they are also both complex systems, where the behavior "of the whole cannot be *predicted* from behavior of parts and complex interactions among parts give rise to *emergent* behavior" [10]. In looking back, complexity may help explain what happened [8], but in looking forward complexity may hinder anticipating what will happen. The behavior of a complex system is defined not only by the elements but the relationships, rules, and processes between elements, particularly as exercised dynamically, where intricate dependencies create unforeseen change and cascading effects. For example, the 787AcqSys was constructed to readily identify issues long before they became problems, by use of a distributed information system across the enterprise. However not all of the 787AcqSys elements used the information system as expected, and subsequently the issue identification behavior was never realized as intended. Turning to the MTOpSys as an example, its behavior is in many ways similar to communication networks which is subject to changes in performance due to network conditions and volume. The objective behavior of the MTOpSys is constructed (partly) in element schedules (arrival and departure times), but when things happen (traffic accidents, detours, police activity, equipment failure), the objective behavior may not be attainable.

Given that complexity gives rise to emergence, and emergence leads to unexpected and unpredicted conditions, such as disruptions, what are the indicators of complexity that suggest resiliency is required? For the two exemplar systems analyzed here, they both share two drivers of complexity: human agents as integral system components and a systems-of-systems (SoS) structure.

- They both rely on human agents for execution and operation. Even when constrained by relationships, rules, and processes, human behavior can be difficult to predict, routinely exhibiting emergence. On the one hand, this may further exacerbate complexity and emergence, potentially leading to unintended behaviors. In the 787AcqSys case, contracts between system elements created unintended behaviors when human agents decided to pursue courses of action that were in the best interest of the element system (home organization) rather than the whole system [7]. However human agents are also in general able to readily demonstrate adaptability and flexibility, characteristics that enable resiliency. For example, when conditions lead to unintended overlapping coverage within the MTOpSys (e.g., two or more buses on one route arriving at stops at the same time), the drivers may begin to intentionally bypass each other to avoid having unnecessary redundancy at stops.
- They both exhibit system-of-system (SoS) characteristics, as described in the table below (Table 4.1).

These systems were constructed to execute according to a plan, or operate in accordance with some governing rules, but emergence – the realized behavior – is what happens when things happen. Emergence can be desirable (anticipated or unanticipated) or undesirable [11]. When things happen that are desired, such as ad hoc communication between mass transit operators enabling passengers to make tight connections, that is emergence. When things happen that are not desired nor anticipated, such as 25,000 employees of the 787AcqSys who went on strike in 2008 over concerns regarding outsourcing [4], this is also emergence, and disruptive. When desired things don't happen, such as the 787AcqSys information system was not used for issue sharing, this is also emergence, which eventually became disruptive.

Emergence gives rise to events/activities that are not in accordance with the AcqSys plan or OpSys governance, and may be considered opportunities or disruptions. When these events or activities positively impact the performance of the system, they may be considered opportunities; conversely when they adversely impact performance, they may be considered disruptions. It is this latter scenario, where the effect of disruptive events is abated or negated by effort, that is explored to consider engineered resilience.

4.4 AcqSys and OpSys Resilience: A Review

Resilience is a widely used term and in general can be understood as the ability to operate through some adverse condition. Several formal definitions exist such as resilience is "a property associated with system behavior – enabling continued

Table 4.1 SoS characteristics [15] of the AcqSys and OpSys

Ref.	SoS characteristic	787AcqSys applicability	MTOpSys applicability
[2, 12]	Operational independence	Components operate independent of AcqSys. Tier N + 1 AcqSys are separate functioning organizations and exist independent of the 787AcqSys to support other customer needs	Components *can* operate independent of OpSys. Elements of the MTOpSys can be deployed to provide transportation to specified groups (such as charters) rather than the mass public. Elements are operated by separate and distinct agencies
[2, 12]	Managerial independence	Components operate independent of AcqSys. Tier N + 1 AcqSys are separate functioning organizations and make strategic and resource decisions independent of the 787AcqSys	Components operate independent of OpSys. Elements are owned and managed by separate and distinct agencies
[12]	Evolutionary development	Occurred incrementally in accordance with acquisition strategy. Complete 787AcqSys occurred over time with successive contracts let to fulfill service and product needs as required. Restructuring also occurred as the need arose	Occurred over time to satisfy evolving need within constraints. Changes in user demographics, economics, as well as area congestion continually define transportation needs which the system responds to by adding/altering service routes, capacity, and frequency
[12]	Geographic distribution	Integrators and suppliers distributed around the globe	System elements span regional (city/county) boundaries
[1]	Belonging	Evident by elements electing to participate contractually. Tier N + 1 AcqSys elected to participate in the 787AcqSys by responding to contractual offers	Regional leaders enable participation to support a common goal of providing transportation to riders traveling throughout the area. A tangible example of belonging is demonstrated by utilization of "common currency" in the form of transit passes
[1]	Connectivity	Interfaces exist across the enterprise for coordination. 787AcqSys would have inherently had to leverage interfaces vertically between related AcqSys (refer to sidebar of Fig. 4.4), but horizontal connectivity was also likely required	Conjunction points between elements provide enhanced coverage. Service routes overlap and intersect, enabling riders to combine distinct service providers to create tailored routes to meet their needs

useful service in the face of disruptive events" [16]. For example, the 787AcqSys continued to usefully execute despite many disruptions (e.g., supplier failure, labor strike, improperly built parts) and delivered the Dreamliner, but not in accordance with the cost or schedule performance targets. The Dreamliner was 3 years late and greater than \$10 billion over the planned budget [3, 6]. MTOpSys also continue to usefully operate despite disruptions such as infrastructure fires, bomb threats, and unplanned shutdowns, but not in accordance with availability or timeliness performance targets. Until the disruptions (and associated collateral damage) were resolved, which may take anywhere from hours to days to weeks, the system service level will be reduced (relaxed performance targets) because of (1) limited capacity of alternate mass transit means and (2) elongated travel timelines necessary to go around disruptions. Using this definition, it can be said that any AcqSys or OpSys that is not terminated but continues to provide some value despite adverse conditions is resilient. The 787AcqSys and MTOpSys would be considered resilient according to this definition because they continued to exist (resourced through disruption $effort_{td}$ and $effort_{r}$) and they were able to alter performance targets ($capability_{td}$).

Another explanation of resiliency is that it "*comprises planning, behaviors, hardware, and software that enable a system to continue providing useful service in the presence of disruptions – unexpected, unpredictable conditions – resiliency is 'outward' looking*" [14]. This definition is similar to the previous and as such both systems would be considered resilient as they continued to provide useful service through disruption. However this definition also includes a description of disruption as being unknowable and external. Did these systems experience disruptions of this kind? That question cannot be answered here, but what can be said is that if such disruptions occurred, these systems still continued to provide useful capability. Some disruptions that appear unexpected or unpredictable should not be, because even though they may seem to occur suddenly, in some cases conditions that enable disruptive events exist for some period of time and go unnoticed or unattended to until a disruption triggers a failure.

For the 787AcqSys, it could be argued that the disruptive examples described here should not have been unexpected or unpredictable. For example, significant time elapsed while suppliers were making parts that ultimately would not fit together. Had this process been an item of concern, then earlier awareness of the growing problem would have enabled pre-emptive corrections sooner. This is not to say that a disruption would have been avoided totally; rather that earlier awareness and action might have afforded a greater trade space of options to mitigate the consequences, possibly reducing the $effort_{td}$ and the $effort_{r}$ required to deliver the Dreamliner. Similarly, the example disruptions of the MTOpSys were not necessarily unexpected or unpredictable, even though they may have appeared to occur without notice. For example, after a serious train tunnel fire in 2015 in the Washington DC metro, additional tunnel fires such as those that occurred in early 2016 should not have been unexpected because the conditions that caused the fires (frayed cable jumpers) still existed. Earlier proactive measures may have reduced the need for $effort_{td}$ and $effort_{r}$ (complete 29 h shutdown of the metro system to inspect) to continue mass transit operations. It's also quite possible that earlier proactive measures may have required greater resources (than $effort_{td}$ and $effort_{r}$) for train operations, which basically means that altered performance targets ($capability_{td}$) were an enabler of resiliency.

Yet another definition of resilience is "*the capability of a system with specific characteristics before, during and after a disruption to absorb the disruption, recover to an acceptable level of performance, and sustain that level for an acceptable period of time*" [3]. It is worth noting that this definition required clarification of several terms. Using those clarifications, the definition could be rewritten as: "the capability of human-made systems [systems], enabled by design properties (such as redundancy) or enabled by processes (such as corrective action) [specific characteristics], to anticipate [before] and survive [during] an internal or external event that initiates a sudden or sustained performance reduction [absorb]; and recover from that performance reduction to an acceptable level of performance for some determined long-term period of time." Once again, using this definition, it can also be said that unless terminated, the 787AcqSys and MTOpSys were resilient as previously explained. This definition, however, does acknowledge that a system can be viewed as being in a "state" relative to a disruption (before, during, and after). It does not however specifically call out another state that may be difficult to detect, which will be referred to as "drift."

When the AcqSys execution diverges as planned, or the OpSys is executed in manners inconsistent with the governance, the system can be described as being in "drift." Drift is a condition where variation from the objective or intended exists, possibly unnoticed. "Failure *[disruption]* occurs when an error *[drift]* reaches a system boundary *[is noticed]*" [14]. The 787AcqSys was disrupted when the supply chain set up to design, manufacture, and integrate parts failed to deliver parts that fit together because there was no explicit plan for blueprints like those that would have been normally prepared for a supply chain [9] – variation from the intended. If the lack of a common blueprint, or part incompatibility had been noticed earlier, $effort_{td}$ and $effort_{r}$ to create parts might have been mitigated or avoided. The MTOpSys was disrupted by tunnel train fires at least 2 times post the 2015 incident, due to continued deterioration of cable jumpers – variation from the intended. If the action had been taken earlier, particularly after the 2015 incident, then the $effort_{td}$ and $effort_{r}$ for mass transit (work around for subsequent fires and the shutdown) may have been mitigated or avoided.

4.5 Engineering and Measuring Resilience

The discussions above regarding resilience and the impact of disruptions on the example systems offer insight into engineering and measuring resilience. Consider resilience, a property of the system that determines what happens when disruptions occur, as the measurable effort and capability in response to disruptions. That response, a set of actions or steps taken to continue/restore a capability, is a process. An approach to create a resilient system begins with a concept of operations (CONOPs) for resiliency, as defined by a process.

4.5.1 A Concept of Resilient Operations

Resiliency as a process includes preparing strategies and tactics, employing means and methods, and learning from experience to handle disruptions. This process can be viewed as a series states and associated activities to deal with disruptions. This process is often described temporally relative to the disruption. For example, the following five time periods (long-term prevention, short-term avoidance, immediate-term coping, cope with ongoing trouble, and long-term recover) [13] can be mapped into a series of states – before, during, and after (BDA) – that may include functions and methods such as those listed in the table below.

STATE	May include functions	Aided by methods such as
Before	Anticipate, prepare, prevent, avoid, monitor	Risk management Critical process review Lessons learned adoption
During	Survive, cope, recover, monitor	Reduced performance targets Alternate (backup) methods
After	Restore, rebuild, adapt, monitor	Reset/reestablish performance targets in reconstituting the disrupted capability

This list of functions/methods is not meant to be exhaustive, but exemplary. The functions are self-explanatory, but the purposes of the methods are explained below:

- Risk management (RM) – includes the identification, analysis, and management of risks to prevent a disruption from occurring or mitigate the severity of a disruption. However as stated earlier due to complexity, the ability to anticipate and therefore manage all potential disruptions becomes untenable. Resiliency is complementary to RM, providing an ability to respond to disruption that could not be prevented, mitigated, or predicted.
  - If the 787AcqSys supply chain uncertainty (which existed until parts integration was demonstrated) had been managed as a risk, it's possible that this disruption would have been avoided or the impact reduced.
  - If the MTOpSys operations included looking for uncertainty at train stations (such as items present that should not be), it's possible that the impact of this disruption (bomb threat scenario) could have been reduced.
- Critical process review – includes identification of those capabilities, activities, or events that are of significant importance that they must be assured or supported by alternate means should the process be compromised ("plan b").
  - If "building compatible parts" had been viewed as 787AcqSys supply chain critical processes, then either additional effort could have been put into assuring this process, and/or an alternate set of suppliers ("plan b") could have been leveraged to avoid or reduce the impact of the disruption.

  - If the MTOpSys train operations critical process included safe passage through train tunnels and stations, then additional effort could have been spent to assure safe conditions to minimize the impact of disruptions.

- Lessons learned adoption – includes incorporating the experiences of prior disruptions (or disruptions in other but relevant systems) as a means of avoidance, prevention, or mitigation.
- Reduced performance targets – a degradation to the measurable performance of some function or capability. In the face of disruption, this may be the "plan b" or a coping strategy.
  - The 787AcqSys supply chain had no choice but to change the plan (delivery dates, integration timetables) as there was no alternative capability to produce parts in place.
  - The MTOpSys experienced schedule delay due to tunnel fires and suspicious objects at station. These delays were not only for those passengers who implemented a "plan b" (using other transportation means) but also for those who waited out the disruption and remained in place on the train until service was restored.
- Alternate (backup) methods – a method of achieving the desired capability via a system independent of the disrupted system ("plan b").
- Reset/reestablish performance targets – a return to previous targets, or the establishment of new performance targets once the disruption has been mitigated.
  - The 787AcqSys supply chain established new performance targets (revised plan).
  - The MTOpSys reset performance targets (adjusted schedule) while the lingering impact of the disruption was being addressed, and then returned to normal performance targets (existing schedule) once the disruptions were resolved.

A CONOPs for resiliency built around the states of before, during, and after with associated functions and methods can be used as a foundation for establishing and allocating functionality necessary to engineer resilient properties in systems (culture, processes, and assets) .

4.5.2 Measuring the Effort and Effect of Resilience

Valid, consistent, and widely used methods to measure resiliency are not readily available. Thus researchers are creating their own. For instance, the "non-existence of a reliable and valid scale measuring organizational resilience" was the motivation for creating a "scale of organizational resilience construct" [7], which resulted in three dimensions (robustness, agility, and integrity) of a resilience construct.

Researchers are also reshaping how to think about and thus measure resilience. "Traditional planning has viewed the crisis plan as an outcome of a process to be utilized in a step-by-step fashion during a crisis. . .suggests a new paradigm, one that focuses on creating organizational structures and processes that build organizational resilience potential. The objective is to develop a scale to measure latent resilience in organizations" [17].

For the purposes of engineering resilience, there are at least two reasons to measure resilience.

1. To perform a cost-benefit analysis: Is the effort associated with building in resilience worth the benefits of resilience? This would entail not only quantifying the "cost of resilience" but also the value of what resilience provides. For the same reasons that it is difficult to predict the behavior of complex systems (what happens when things happen), it is similarly difficult to predict the behavior of resilient complex systems (what happens when disruptive things happen). Thus putting a value on resilience is challenging.
2. To compare between alternatives: Which among the competing resiliency options or courses of action is the preferred selection? Such a comparison would require quantifiable metrics for each option which can be at a minimum relatively ranked. A resilience process built around the before, during, and after CONOPS, as well as the measurable effort and capability in response to disruptions, is proposed as a framework to quantify the effort and effect of resiliency.

4.5.3 A Framework for Measuring Resiliency

Figure 4.1 depicts a resilience process that includes three sequential states BDA and the condition drift. The arrows indicate events (disruption, recover, restore/reset) that mark the transition from one state to another. Drift, a measure of variance from the expected/planned, exists independent of the states, but whenever the variance causes a disruption, the system will transition into the during state. This figure is two-dimensional, but in reality, a system can experience many disruptions concurrently; therefore every time a disruption occurs, a new BDA wheel cycle is instantiated (creating a three-dimensional state model – not depicted here). Accordingly, a system can be in many BDA states concurrently. Finally, while the BDA states are drawn to equal size in the figure below, it's plausible that greater latent resilience may enable reduced effort associated with during and after.

Drawing on the previous discussion regarding effort and capability and a resilience CONOPs, the following definitions are offered. The *effort* to provide a *capability* when faced with disruptions changes to the $effort_{td}$ to provide a $capability_{td}$, and the $effort_r$ to reinstate a *capability*. The additional effort ($effort_{td}$ and $effort_r$) reflects an increase in scope or work and is annotated as $\text{State}_{\text{mag}}$. This additional effort ($\text{State}_{\text{mag}}$) may also require changes to timelines and is annotated as $\text{State}_{\text{dur}}$. Applying these terms to the BDA states, the following metrics are proposed as a tool for engineering resilient systems:

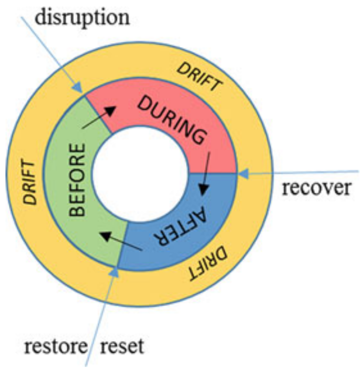


Fig. 4.1 Resilience process

- B_{Mag} → ideally is zero as this need not be considered "*additional effort*" but rather part of the intended effort to create a resilient system. This is work that should be done independent of a disruption, except in cases where a disruption results in learning/adapting that updates the functions and/or methods of before.
- B_{Dur} → is the time between after and during and ideally is greater than D_{Dur} and/or A_{Dur} (meaning as disruptions are mitigated or minimized and less time is spent in during and after).
- D_{Mag} → is the increased scope or work to survive, cope, and recover from a disruption. In general the objective would be to minimize the increased scope or work, while balancing near- and long-term performance objectives.
- D_{Dur} → is the extra time required to survive, cope, and recover from a disruption. In general the objective is to shorten this timeline as much as possible.
- A_{Mag} → is the increased scope or work to restore, rebuild, and adapt post disruption. In general the objective would be increased efficiency over time, in executing the increased scope or work, to restore, rebuild, and adapt.
- A_{Dur} → is the extra time required to restore, rebuild, and adapt post disruption. In general the objective is to shorten this timeline as much as possible.

No metric is defined for drift since the scope of work that needs to be accomplished (monitoring and review) is continual and independent of disruption, and ideally a part of the normal system function/operation. This is not to say that drift does not consume resources; in fact, critical decisions regarding the structure and performance of the monitoring/review function are cost/benefit consideration.

The above set of metrics provide a framework for engineering and measuring resiliency. However the literature is replete with problems associated with metrics; therefore this set is subject to the same challenges as any set of metrics (such as measurement efficacy, changing behavior by the introduction of metrics, not capturing outcomes, etc.). Further investigation of the methodologies proposed here is needed.

4.6 Observations Regarding Engineered Resilience

Organizations seeking to infuse resiliency into their systems as a way to deal with disruptive events and circumstances will find that just like other system qualities, resiliency is a design objective. The objective may apply to many parts of a system including culture, processes, and assets. A starting point is to decide exactly what resiliency means for the mission and system under consideration. Although only a few definitions were discussed here, there are many. These definitions imply a process on which resilience depends. In order to design for resilience, it must first be understood, in context. A way to build that understanding is to leverage a resilience process to develop a concept of resilient operations, from which system functions and metrics can be derived.

Resilience is a widely used term and in general is understood as the ability to operate through some adverse condition. Part of understanding what resiliency means for the mission and system under consideration is to understand the reason/need for resiliency. For example, the two example systems discussed here had similar needs: to better manage disruptions (unavoidable conditions, unmitigated consequences, or unforeseen circumstances) and therefore minimize the need for increased resources ($effort_{td}$ and $effort_r$) to deliver a product or provide service in the face of disruption. The nature of the systems was very different however (creating a product vs providing a service), and the difference is important for considerations related to the resilience process. The AcqSys created a product, and the OpSys provided a service repeatedly. The opportunity to increase the efficacy of the resilience process may be enhanced in situations where repetition is involved (mainly because opportunities to learn and adapt are greater). Therefore, the nature of the system should drive the resilience process and CONOPs. In an increasingly complex world, where systems of systems provide functionality to enable operational capabilities, engineered resilience has the potential to facilitate effective handling of change, events (internal vs external), or conditions that suddenly or, in a sustained fashion, adversely impact system behavior and performance.

Appendix A: The Boeing 787 Acquisition System

In 2003 Boeing announced their plans to build the 787 Dreamliner in response to declining sales. In an effort to shorten the development and production cycle time and cost, Boeing decided to use a different supply chain approach. Their previous approach (Fig. 4.2) employed Boeing as the integrator directly interfacing with a plethora of suppliers, but their new approach employed Boeing as an integrator of integrators (Fig. 4.3). Ultimately, this new approach contributed to the poor program performance of the 787AcqSys, which failed to achieve the development cost and schedule targets Boeing sought. Boeing had to re-architect the approach along the way, and in some cases return their prior supply chain structure.

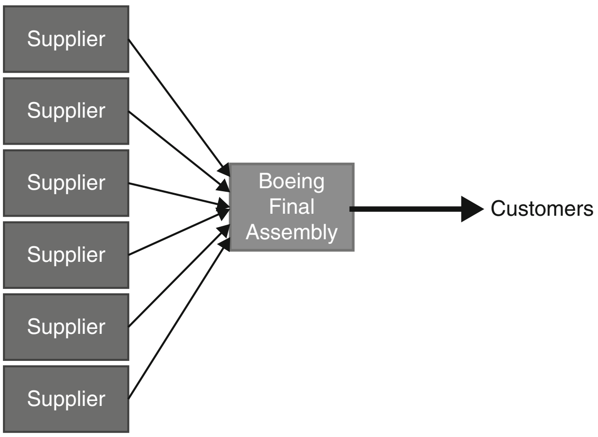


Fig. 4.2 Traditional Boeing airplane manufacturing supply chain source [18]

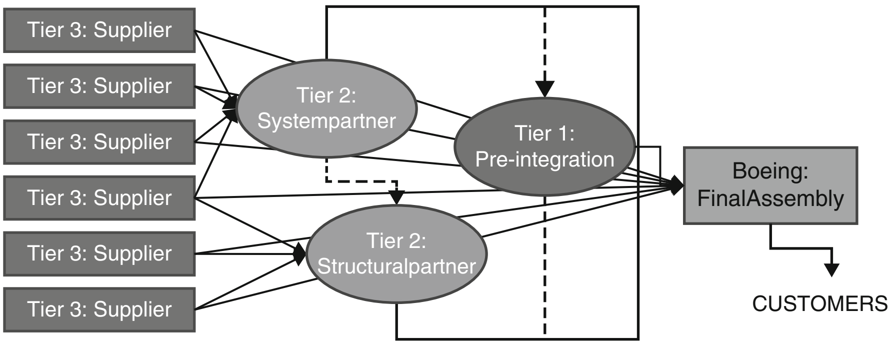
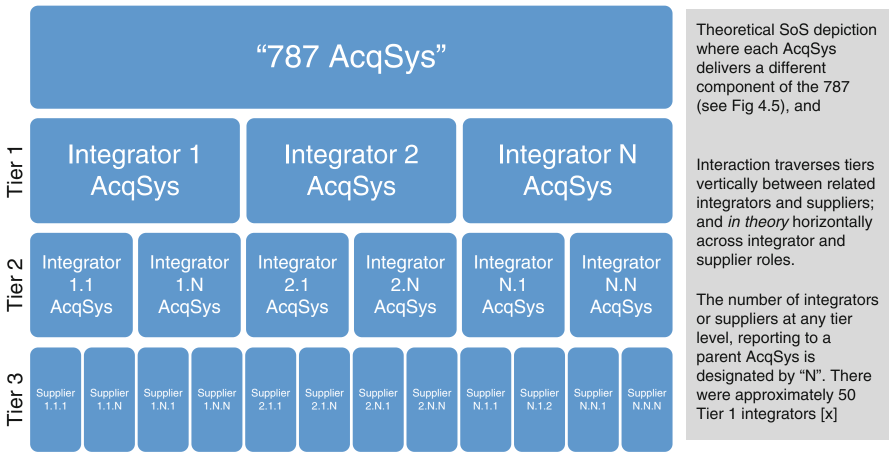


Fig. 4.3 Redesigned Boeing airplane manufacturing supply chain source [18]



Theoretical SoS depiction where each AcqSys delivers a different component of the 787 (see Fig 4.5), and

Interaction traverses tiers vertically between related integrators and suppliers; and *in theory* horizontally across integrator and supplier roles.

The number of integrators or suppliers at any tier level, reporting to a parent AcqSys is designated by "N". There were approximately 50 Tier 1 integrators [x]

Fig. 4.4 "787AcqSys SoS"

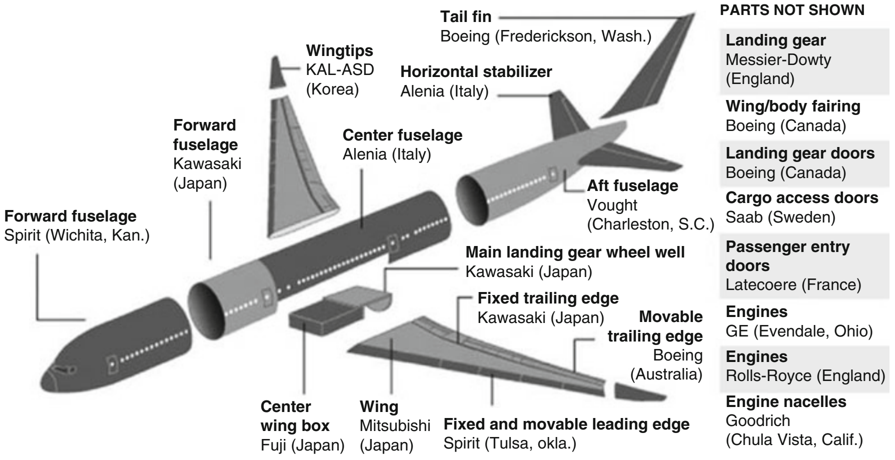


Fig. 4.5 Dreamliner subassembly plan source [18]

References

1. Boardman J, Sauser B (2006) System of Systems – the meaning of. Proceedings of the 2006 IEEE/SMC international conference on system of systems engineering, April 2006
2. Cureton K (2014) Lecture #1: "Syllabus, Definitions & Characteristics", SAE 599: Resilient, Cyber Secure Systems & System-of-Systems, Faculty of Engineering, USC
3. Denning S (2013) The Boeing debacle: seven lessons every CEO must learn. Available at: http://www.forbes.com/sites/stevedenning/2013/01/21/what-went-wrong-at-boeing/. Accessed 10 Jan 2015
4. Denning S (2013) What went wrong at Boeing? Forbes.com, Available at: http://www.forbes.com/sites/stevedenning/2013/01/21/what-went-wrong-at-boeing/. Accessed 10 Jan 2015
5. INCOSE (2015) INCOSE Resilient Systems Working Group Charter. Available at: http://www.incose.org/docs/default-source/wgcharters/resilient-systems.pdf?sfvrsn=6. Accessed 26 Apr 2015
6. INCOSE Systems Engineering Handbook (INCOSE) (2012) Definition of life cycle stages. Available at: http://sebokwiki.org/wiki/System_Life_Cycle_Process_Models:_Vee. Accessed 18 Apr 2016
7. Kantur D, Iseri-Say A (2015) Measuring organizational resilience: a scale development, İstanbul Bilgi University, Turkey and Bogazici University, Turkey, 10.17261/Pressacademia.2015313066. Available at: http://dergipark.ulakbim.gov.tr/jbef/article/view/5000150856. Accessed on 18 Apr 2016
8. Keskinen A, Aaltonen M, Mitleton-Kelly E (2003) Organisational complexity, Finland Futures Research Centre, Turku School of Economics and Business Administration, Available at: https://www.utu.fi/fi/yksikot/ffrc/julkaisut/tutu-julkaisut/Documents/Tutu_2003-6.pdf. Accessed 12 Apr 2016
9. Lewis R (2015 Spring) Term paper – "Acquisition systems: an initial look from a system/system of system integration perspective, SAE 548: System/System-of-Systems Integration, USC
10. Madni A (2016) Lecture #1: "Class overview", SAE 599: Engineered Resilient Systems, Faculty of Engineering, USC

11. Madni A (2016) Lecture #4: “System architecture: key aspect of resilience”, SAE 599: Engineered Resilient Systems, Faculty of Engineering, USC
12. Rhodes D, Ross A (2014.) Engineered resilient systems – systems engineering: knowledge capture and transfer technical report, SERC-2014-TR-045-1, Massachusetts Institute of Technology, Available at: http://www.sercuarc.org/wp-content/uploads/2014/11/SERC-2014-TR-045-1-Engineered-Resilient-Systems-RT-103.pdf. Accessed 12 Mar 2016
13. Sheard S (2008) A framework for system resilience discussions, Stevens Institute of Technology, Third Millennium Systems, Available at: http://seir.sei.cmu.edu/sheard/SheardSysResilience.pdf. Accessed 12 Apr 2016
14. Sievers M (2016) Guest lecture – “A Brief Introduction to Fault-Tolerant Systems & Engineered Resilience Spring”, SAE 599: Engineered Resilient Systems, Faculty of Engineering, USC
15. Sievers M, Madni A (2015) Lecture #2: “Definitions, Examples, and Failures”, SAE 548: System/System-of-Systems Integration, Faculty of Engineering, USC
16. Sievers M, Madni A (2015) Lecture #4: “SI/SoSI Ontology Engineered Resilient Systems”, SAE 548: System/System-of-Systems Integration, Faculty of Engineering, USC
17. Somers S (2009) Measuring resilience potential: an adaptive strategy for organizational crisis planning. J Conting Crisis Manag 17(1):12–23. Available at SSRN: http://ssrn.com/abstract=1394697 or http://dx.doi.org/10.1111/j.1468-5973.2009.00558.x. Accessed on 18 Apr 2016
18. Tang C, Zimmerman J (2009) Managing new product development and supply chain risks: the Boeing 787 Case, Supply Chain Forum Int J 10(2). Available at: http://bus545-boeing.wikispaces.com/file/view/Boeing+787+Case.pdf. Accessed 11 Apr 2015
19. Tang C, Yeh B, Zimmerman J (2013.) Boeing’s 787 dreamliner: a dream or a nightmare, UCLA Anderson Global Supply Chain Blog, Available at: http://blogs.anderson.ucla.edu/global-supply-chain/2013/05/boeings-787-dreamliner-a-dream-or-a-nightmare-by-christopher-tang-based-on-work-with-brian-yeh-pwc-advisory-and-joshua.html#sthash.Se9G8tcs.dpuf. Accessed 27 Apr 2015

Chapter 5
High Reliability Imperative for Autonomous Networked Vehicles

Allen Adler and Azad M. Madni

Abstract Autonomous vehicles need high reliability for consumer acceptance. A high-reliability system is capable of relatively error-free operations over extended durations making consistently good decisions that result in highly reliable and safe operations. High reliability is an imperative for autonomous networked vehicles. This paper reviews currently available approaches for developing high-reliability systems. We pose the question whether reliability requirements for self-driving vehicles should be similar to those for other high-reliability systems or should we draw on and extrapolate the body of knowledge for human-operated systems. We provide an analysis that helps with answering this question.

Keywords Autonomous systems • Self-driving vehicles • System of systems • High-reliability systems

5.1 Introduction

There is a rapidly growing body of evidence that driverless cars can perform the basic functions necessary to drive a car. However, there is no way to assure that driverless cars can operate safely in all conceivable situations. Thus, driverless cars face a challenge that is similar to the one faced by aviation and electric power, the need to assure an extremely low risk of catastrophic failure. The aviation community has employed system engineering to achieve an extremely low and continually decreasing risk of failure. Today system engineering can potentially play an important role in assuring an acceptably low risk of catastrophic failure in driverless cars.

Recent progress in driverless cars has been largely enabled by several key advances in computing and artificial intelligence. However, it is unlikely that an

A. Adler (✉)
The Boeing Company, Chicago, IL, USA
e-mail: allen.adler@boeing.com

A.M. Madni
University of Southern California, Los Angeles, CA, USA
e-mail: Azad.Madni@usc.edu

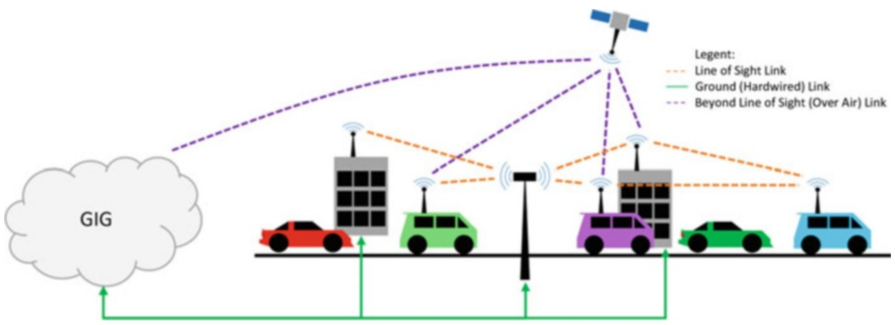


Fig. 5.1 System of systems (SoS) network of AVs linked through a smart infrastructure. Note the presence of two legacy vehicles in the environment which are not members of the SoS (Source: Madni [6], used with author's permission)

acceptable level of safety and reliability of driverless cars can be assured by treating driverless cars exclusively as software systems [1] (Ref: Koopman and Wagner). Rather, driverless cars must be viewed as part of a much larger system of systems (SoS) network. In fact, today, driverless cars are being designed to be network enabled. An autonomous vehicle network is essentially a system of systems (SoS) that is amenable to analysis using SoS analysis methods.

Figure 5.1 shows multiple autonomous vehicles (AVs) interconnected via a smart infrastructure enabled by the "Internet of things" [2]. Each vehicle is an independent system that leverages the connectivity provided by the smart infrastructure to communicate and share data with other vehicles and structures. The AVs are members of an AV network, which has the properties of a system of systems (SoS) enumerated in Table 5.1. Recognizing an autonomous vehicle network as a SoS leads to the methods developed for SoS analysis by Madni et al. [3, 6] and discussed in [4, 5]. These authors outline a logical progression from SoS and system use cases to system and stakeholder objectives and finally to SoS and system requirements. Collectively, these considerations impact both SoS integration and SoS verification and validation (V&V).

Anticipating rapid adoption of AV technology, we can expect that this SoS will include legacy (i.e., human-driven) vehicles. The transition from the current state to a fully autonomous AV network is likely to occur in stages. Prior to all vehicles becoming fully autonomous and networked, there will be a transition period in which networked AVs will coexist with human-driven standalone vehicles. This stage in the evolution of networked AVs is also depicted in Fig. 5.1.

This paper takes a critical look at driverless cars as a SoS network and discusses using examples how this approach can help with the design, verification, and continuous improvement of safe AV networks.

Table 5.1 An AV network is a SoS

Operational independence of AVs
AVs operate independently as part of a traffic ecosystem
Managerial independence of AVs
AVs governed independently while being part of traffic ecosystem
Evolutionary development of SoS
Development and existence is evolutionary with functions and purposes added, removed, and modified with experience and need
Emergent SoS behavior
AV-SoS performs functions and carries out purposes that do not reside in any single AV
AV-SoS behaviors are emergent – cannot be realized by a single AV
Geographic distribution
AVs primarily exchange information – not mass or energy

5.2 Safety and High Reliability

We cannot begin to consider the design of a driverless cars SoS network, without a clear idea about the level of safety and reliability that is required. It would certainly be unacceptable to have a fatality rate above the current rate of 1 fatality per 100 million miles traveled [7]. Human error is responsible for a significant fraction of those fatalities. For example, 30% of the fatality rate is attributed to alcohol-impaired driving [7]. This raises an important question about the reliability requirements for driverless cars. Should they be based on experiences with human-driven cars or on the lowest levels that could be achieved with a properly designed SoS? The difficulty of this question is underscored by recent experiences. The number of fatalities caused by faulty Takata airbags was three orders of magnitude smaller than the number of fatalities experienced in the USA in 2015 [8]. As appropriate, there have been massive recalls costing billions of dollars to correct these faulty airbags. Clearly, the tolerance for human error is much higher than for mechanical error. Accordingly, we could reasonably expect that the fatality rate limit required for driverless cars could be orders of magnitude below the rate experienced with human drivers. On the other hand, it could be argued that achieving a fatality rate with driverless cars that was 10% lower than the current rate would result in saving thousands of lives. It is well beyond the scope of this paper to propose a resolution of this dilemma, but until it is resolved and a maximum tolerance level for fatal faults is established, an effective SoS design cannot be undertaken.

Designing for and assuring a high level of reliability is fundamentally a matter of dealing with the unknown. Table 5.2, which is meant to be illustrative but not exhaustively enumerate, lists some of the techniques that are currently used to engineer high-reliability SoS. Each technique approaches the unknown differently. More generally, we can envision three approaches to dealing with the unknown. First, we could restrict the realm of operations to minimize unknowns. For example, if a car were restricted to operate on a track in a completely controlled space, then a very high safety standard could be met. An example of this is the automated train

Table 5.2 Approaches for high-reliability systems

Analysis using physics-based model
For example, the lifetime of a solid state junction is based on solid state diffusion of dopants in a semiconductor
Analysis using end-to-end models of large systems
To derive a system failure rate from component failure rates
Setting safety margins
Adopting more stringent requirements for key performance parameters to compensate for unknowns
For example, a system that is required to last 10 years may be designed to last 20 years
Redundancy
Functional redundancy – achieving redundancy using dissimilar methods
Physical redundancy – achieving redundancy using identical hardware
Testing
Over the range of expected operating conditions
Beyond the range of expected operations
Manufacturing process control
To ensure that what is built is the same as what was tested
Operational limits
That preclude the system being operated outside of the envelope that has been tested

systems that carry passengers between airport terminals in some airports. Secondly, we could overdesign to account for unknowns. Structures are often designed to 1.5× or sometimes 4× the actual loads expected, thereby accommodating the lack of perfect knowledge of the loads that will actually be encountered. A third approach is to purposefully explore unknowns with test programs. Each approach can have an impact on how the SoS is designed and how it is operated.

5.3 System of Systems (SoS) Architecture

The reliability of driverless cars will depend heavily on how they interact with the external environment. We will consider both interactions with the physical environment and interactions with the information environment and mention some of the key choices that will define the architecture of the AV network.

5.3.1 Physical Architecture

In thinking about possible physical architectures for the AV network, it is useful to consider how the airspace is regulated. Regulating authorities around the world divide the accessible airspace into different regions having different levels of control. This airspace architecture has contributed to the steady decline in aviation accidents experienced over the last half-century [9]. Many of the basic concepts underpinning this architecture have been compatible with advances in technology including the advent of uninhabited air vehicles (UAVs).

Of course, the roadway system does have some elements of control. Not all drivers can operate all vehicles, and not all vehicles can be driven on all roadways. However, careful design of a layered system with rules for access and operation of driverless cars could help significantly with the design and validation of highly safe and reliable driverless cars. As is the case for the airspace architecture, the fundamental trade-off is between the benefit resulting from reducing the risk of a fatal collision and the cost associated with reducing the utility of a driverless car.

Another benefit of a layered control system is its compatibility with a gradual transition to driverless cars. It would also facilitate experimentation with a variety of operational concepts. Exploration of these physical architectures warrants significant attention. The methods of SoS design, particularly various types of modeling and simulation techniques, are well suited for this purpose.

5.3.2 Information Architecture

Safe operation of a driverless car depends on enhanced situational awareness. In addition to basic functions such as navigation and vehicle trajectory control, driverless cars need to avoid all collisions without experiencing unacceptable delays caused by having to respond to false alarms. In short, driverless cars must be able to detect, identify, and track all objects that could be involved in a collision and take appropriate collision avoidance action.

For decades there has been a massive amount of research and development of intelligent sensor systems for detection, identification, and tracking of objects. Despite massive investments and impressive technical achievements, this problem has not been solved. Recent advances in sensors, particularly LIDARs and millimeter-wave radars coupled with progress in sensor fusion, are grounds for optimism that a solution to this problem for driverless cars is at hand. This solution will comprise onboard and off-board sensors using different phenomena and a computing network to fuse this diverse data all tied together with a high-reliability communications network. This approach will also raise cybersecurity and privacy challenges which must be addressed.

5.4 System of Systems (SoS) Testing

Complex systems often exhibit behavior that was not planned for or anticipated in the original design. Undesirable consequences of such unanticipated behavior can be mitigated by purposefully designing the SoS to exhibit capabilities that exceed the minimum levels needed to satisfy known requirements. For example, the SoS can be required to be tested under conditions that are more stressful than what would be encountered in actual operations. Adding capability, expanding the test regimen, and restricting the envelope of operations are all costly. Perhaps the greatest challenge of the design of a SoS for driverless cars will be to properly evaluate this trade-off between cost and reliability.

An AV system of systems consists of mechanical and electrical subsystems, sensor subsystems, and software subsystems including artificial intelligence subsystems such as deep learning neural networks. Methods such as those listed in Table 5.1 can benefit each of these subsystems. Mechanical and electrical subsystems can be designed and tested using well-known methods once the appropriate requirements have been established. Sensor subsystems typically include software that interprets the output of physical sensors. The expected rates of false negatives and false positives must be evaluated for each sensor subsystem. This requires testing in all conditions potentially encountered in actual operations. Similarly, with supervisory subsystems that are designed around a series of "use cases," there will always be uncertainty about how the system handles a situation that is significantly different from these use cases.

Subsystems based on artificial intelligence, where behavior depends on cumulative experience, will require rethinking of some traditional approaches. Successfully testing high-reliability AVs will require answers to the following questions. First, what is the range of situations (including physical environment, traffic environment, and maintenance conditions) to test autonomous systems? Second, does the concept of "safety margin" apply to autonomous systems based on machine learning? Third, how to test high-reliability systems that exploit deep learning algorithms? If a problem surfaces during test, how will the solution be implemented in other systems if the exact state of the system under test is system specific? High-reliability SoS need to be resilient in the face of external disruptions. Thus, modeling and deep machine learning play an important role in the system's ability to learn from experience and continue to exhibit increasing levels of resilience. The challenge here is to identify appropriate modeling approaches that have the requisite semantics, are scalable, are amenable to verification and validation, and facilitate test and evaluation.

5.5 System of Systems (SoS) Evolution

The SoS envisioned in Fig. 5.1 will not be realized by starting anew and redesigning vehicles, roadways, communication systems, and business enterprises that enable them. Instead, the end state will be reached by evolution from the existing SoS. The improvements achieved in each step along this evolutionary path must be economically justifiable. One of the challenges to realizing any of these improvements will be assuring that an appropriate fraction of value created flows to those responsible for the improvement. In other words, design of the evolving system of system must include business and regulatory considerations. We expect this evolution to progress along the “S-curve” that is typical of technology development with three phases. Each phase merits its own approach to system of systems design.

The first phase of this development will progress simultaneously along three directions. First, individual subsystems that are early applications of driverless vehicle technology will be added to legacy vehicles. This trend has already begun with features such as adaptive cruise-control and real-time navigation. Second, fully autonomous vehicles will be deployed in structured environments such as factories and warehouses. Third, we can expect to see small-scale demonstrations of an AV network. The first two trends will create substantial economic value, but the increased productivity will be a small fraction of what we expect to ultimately achieve. Each of these activities will support advances in modeling and simulation that when combined will greatly advance our ability to analyze large AV networks.

The second phase of evolution would result in the widespread adoption of AV networks. As confidence is gained with the reliability and robustness of these systems, implementation barriers will be overcome in an increasingly more systematic and predictable way. At the same time, costs will continue to fall as economies of scale come into play. This evolution could proceed at a rapid pace reminiscent of the pace of adoption of mechanized vehicles in the early twentieth century [10]. On the other hand, it is too early to rule out unseen problems that could considerably slow this progress. Cybersecurity issues, which still loom on the horizon, are an example of such a problem. In this phase, large-scale SoS models will mature rapidly and contribute to rapid transition to widespread AV networks.

The third phase will follow substantial adoption of driverless cars. The anticipated step change improvements in the ground transportation system will have been realized. Moreover, just as in the case of cars and aviation, the rapid step change will be followed by a long period of incremental improvements. These improvements will span the entire ground transport system of systems and include intermodal connections to aviation and ships. The system of systems models built in the second phase of this development will provide a means of evaluating the utility of proposed improvements.

5.6 Conclusion

We stand at the dawn of the age of driverless cars. However, the promise of an autonomous vehicle network will not be realized until its safety and reliability can be assured. Reliability will be assured through a combination of onboard and off-board technologies and roadway architecture. Thus, the autonomous vehicle network is a complex system of systems whose design and validation will be achieved using many of the system of systems engineering tools developed for aerospace, transportation, and energy systems.

The problem of an autonomous vehicle network presents three challenges to system of systems engineering, which must be addressed in the near future. First, reliability requirements must be established. The required level of safety could range from what is marginally better than that of legacy systems to what could be achieved based on experience in aviation. Second, methods of reliability assurance must be developed for systems using artificial intelligence. Third, economic factors must be incorporated in such a way that each phase of the evolution of autonomous vehicle network must be paid for by the value generated in that phase. In other words, we expect that the development of an autonomous vehicle network will catalyze major advances in the field of system of systems engineering.

References

1. Koopman P, Wagner M (2016) Challenges in autonomous vehicle testing and validation. SAE World Congress
2. Kortuem G, Kaswar F, Sundramoorthy V, Fitton D (2010) Smart objects as building blocks for the Internet of things. IEEE Internet Comput 14(1):44–51
3. Madni AM, Sievers M (2013) System of Systems integration: key considerations and challenges. Syst Engin 17(3):330–347
4. INCOSE (2007) Systems engineering vision 2020. INCOSE-TP-2004-004-02, http://www.incose.org/ProductsPubs/pdf/SEVision2020_20071003_v2_03.pdf
5. The Institute of Engineering and Technology (2011) Formal methods: a factfile
6. Madni AM (2017) Transdisciplinary systems engineering: exploiting convergence in a hyper-connected world. Springer
7. NHTSA – National Center for Statistics and Analysis, August 2013
8. http://fortune.com/2016/10/21/takata-air-bag-deaths/
9. Aviation Safety, Boeing Commercial Airplanes (2016) Statistical summary of commercial jet airplane accidents. http://www.boeing.com/news/techissues/pdf/statsum.pdf
10. Gordon RJ (2016) The rise and fall of American growth. Princeton University Press, Princeton

Chapter 6
Resilience Concepts for Architecting an Autonomous Military Vehicle System-of-Systems

Kurt Klingensmith and Azad M. Madni

Abstract Rapid advances in automation are allowing technology to replace tasks once performed by humans. Prototypes of autonomous military vehicles reveal a future system-of-systems with significant potential to positively enhance military operations. However, such a complex system of multiple autonomous vehicles will surely face perilous situations that, if the system is architected incorrectly, could doom the system. Resilience techniques exist to enable a system to face disruptions and continue operating within a specified manner; this paper explores techniques and ideas for applying resilience to such a system.

Keywords Resilience • Systems • System-of-systems • Autonomous

6.1 Introduction

Military systems have continuously leveraged the benefits afforded by networking normally disparate systems. This came of age with net-centric warfare (NCW) and is continuing forward via cyberspace-enabled operations [1, 2]. Robust networking capabilities and cyberspace enclaves coupled with continual technological advances mean military autonomous vehicle (AV) system-of-systems (SoS) are coming soon to the Department of Defense (DoD). In fact, testing is already underway. In 2014, Lockheed Martin and the Tank Automotive Research, Development, and Engineering Center (TARDEC) demonstrated the Autonomous Mobility Applique System (AMAS), in which a "completely unmanned leader vehicle [was] followed by a convoy of up to six additional follower vehicles" [3]. In addition to maneuvering among each other, DoD AVs will need to operate with

K. Klingensmith
US Army, Monterey, CA, USA
e-mail: Kurt.m.klingensmith.mil@mail.mil

A.M. Madni (✉)
University of Southern California, Los Angeles, CA, USA
e-mail: azad.madni@usc.edu

A.M. Madni et al. (eds.), *Disciplinary Convergence in Systems Engineering Research*, https://doi.org/10.1007/978-3-319-62217-0_6

humans; TARDEC's 30-year strategy includes the "[d]emonstration of unmanned vehicles capable of maneuvering with mounted and dismounted units" as a critical component of its outlook [4]. TARDEC envisions autonomous elements in Army formations serving to "augment and enable [soldiers], while filling some of the Army's most challenging capability gaps" [5]. Complementing this is the Army Research Lab (ARL), who envisions unmanned systems transitioning from a "tool to team member" [6]. Integrating AVs with humans has the ability to significantly transform and improve the capabilities of DoD systems-of-systems. AVs will augment current DoD SoSs, integrating with soldiers and a diverse array of technological systems (to include legacy systems).

The preceding concepts and the DoD's broad scope of operating environments and mission sets guarantee that AVs will encounter disruptions. Diverse terrain and weather, varying communications and cyberspace conditions, contextual changes, legacy interoperability requirements, adversarial systems, and various unknowns may generate disruptive events that could ultimately lead to system failure. Thus, it is imperative that systems architects of AV systems architect for resilience. The scope of this paper will focus on adding resilience to the AV system-of-systems (SoS) architecture in a DoD, Army-specific setting. Note that maneuverable Army units and their constituent AVs may form mobile, ad hoc "mesh networks" [7]. Network reliance and an amorphous network composition necessitate exploring the need for network and cyber resilience. Additionally, integrating with humans requires interfaces and behaviors for the AVs that foster resilience in a chaotic and unpredictable environment. This paper will explore these resilience domains and how to architect them into the system.

6.2 The AV System-of-Systems and Resilience

6.2.1 The System

Grouped AVs form a system-of-systems (SoS) as per the DoD, who defines an SoS as "a set or arrangement of systems that results when independent and useful systems are integrated into a larger system that delivers unique capabilities" [1]. In normal situations, a military SoS is typically a directed SoS, in which the "component systems maintain an ability to operate independently, but their normal operational mode is subordinated to the central managed purpose" [1]. Each AV forms an independent system that can serve a purpose on its own but, when massed together or partnered with pre-existing military formations, AVs form an SoS with the potential for emergent behavior [8].

The benefits and objectives of replacing soldiers via an AV SoS are straightforward. As Dr. Azad Madni surmised, "Humans are creative" and adaptive, though "[a]daptivity is a unique human capability that is neither absolute or perfect" [9]. Furthermore, human performance follows the Yerkes-Dodson law, which states that human performance increases with stress and arousal until hitting a peak or optimal point after which additional stress and arousal decrease performance [10]. An AV SoS will operate in numerous contexts under various mission sets, which "can range from non-kinetic operations," such as humanitarian missions, to "kinetic operations and conflict with near-peer states" [11]. Such military operations typically explore the full range of stress and arousal, which in turn could lead to human error, resulting in disruptions. AVs would remove the function of driving from soldiers, thus eliminating a domain in which human error could lead to potentially fatal or mission-altering disruptions. Civilian autonomous vehicles may allow for a rethinking of the dominant architecture, giving way to a "light-weighted [design] as crash risks fall dramatically" [12]. Military AVs may allow for a similar repackaging, increasing occupant survivability and improving other functions of the platform by better employing humans (e.g., weapons systems employment, reconnaissance, or observation and sensing). Finally, the net-enabled aspect of an AV SoS and its automated sensors will allow a more rapid sharing of contextual, sensory information than is currently allowed by human drivers relaying information.

6.2.2 Resilience

The previous benefits and objectives are dependent upon the AV SoS maintaining a certain level of performance. Architecting for resilience becomes imperative given the scalable scope and the high complication and complexity of such an SoS. Several definitions for resilience exist. Resilience is defined "as robustness that is achieved through thoughtful, informed design that makes systems both effective and reliable in a wide range of contexts" [13]. Previously, Madni identified resilience as a capability that allows "a system to circumvent, survive, and recover from failures and ultimately achieve mission objectives" [14]. Additionally, "A resilient system is able to reason about [its] own/environmental states" [14]. A resilient system also has the following ability: "avoiding, absorbing, adapting to, and recovering from disruptions" [15].

The DoD definition specifies the following characteristics of a resilient system, which exhibits "broad utility"; can "repel, resist, or absorb" disruptions; and can "adapt" and "recover" after surviving a disruption [11]. But while undergoing disruption, a resilient system exhibits "graceful degradation" of capability and conducts "real-time trade-offs" during the disruption to manage its state and posture itself for an improved future state [14]. Resilience is a "compound quality attribute," relying on the presence of architectural flexibility (for "expected change") and adaptability (for "unexpected change") [14]. Summarizing from these definitions, resilience allows a system to maintain an acceptable level of performance so as to satisfy its stakeholder objectives despite facing a disruptive, volatile environment. These concepts contributing to resilience are aggregated below:

- *Resilience* – a compound attribute that subsumes flexibility and adaptability [14]; the ability to withstand disruptions, maintain acceptable performance, and continue as a viable system
- *DoD resilience* – broad utility, the ability to repel, resist, or absorb, the ability to adapt, and the ability to recover [11]

To architect for resilience requires some degree of knowledge regarding the system's potential use cases, contexts, and sources of disruption. There are two main types of disruptions: external and systemic [15]. Elements and circumstances beyond the system boundary result in external disruptions [15]. In the case of the AV SoS, consider the diverse terrain a vehicle may encounter in a country such as Afghanistan. Urban roads give way to rural roads and mountain passes, complicated by a variety of weather effects, road types, traffic patterns, and object density. Each shift in context also comes with unique disruptions. External human agents on foot or in their own vehicles may behave unpredictably, resulting in mission-impacting disruptions. Environmental changes, such as falling rocks or decaying mountain roads, may create unexpected restrictions in movement. Weather and light effects may impede movement and restrict the options available to the system, funneling the AVs toward a disruption. Urban infrastructure, mountainous terrain, or weather effects may interfere with communications equipment. In all the environments, the possibility of enemy agents and systems threatens to disrupt. These disruptions could be simple, such as an obstacle that blocks a road and limits movement, or the disruptions could grow in seriousness, to include improvised explosive devices (IEDs), coordinated attacks, or other direct kinetic actions that intend to disrupt via destruction.

Systemic disruptions differ in that they disrupt a "function, capability, or capacity" of the actual system [15]. The harsh conditions of a dirt road could stress a mechanical component in a vehicle to the point of breaking, reducing the function or capability of the vehicle [15]. The supporting network could fail, resulting in the AV SoS losing its net-enabled capabilities. Driving requires that a human driver appropriately interpret the context and act accordingly; for AVs, this would occur via hardware and software sensor failure. Human agents integrated into the SoS may cause systemic failure as they can introduce errors or negatively influence system behavior [15, 16]. Consider the concept of "risk homeostasis," in which human agents will offset a decrease in risk in one area by accepting additional risk in another [16]. In the case of an AV SoS, such a system may reduce risk of vehicular accident due to automated driving, which then results in commanders planning missions that push the system into even more challenging driving situations that test the limits of the automation's functionality and mechanical abilities, resulting in systemic disruptions.

Note that these discussed disruptions are speculative possibilities. Given its military use, a complex AV SoS faces "uncertainty that can propagate in unexpected ways" [14]; furthermore, it will operate in current and future environments, some known, many unknown. When looking at architectural options, "[in] foresight, many pathways appear to lead to an objective, while in hindsight only one pathway appears probable" [17]. This ties into architecting for resilience, in that not all future disruptions, use cases, or contexts are knowable, yet the inevitable disruption will seem obvious in hindsight. Systems thinking may help narrow the scope of possible futures (and option sets) by enabling architects to envision the holistic system, environment, and system experience in such a way that allows foresight on potential change cascade triggered by future use cases [18].

Systems thinking also prods the architect toward interdisciplinary and even transdisciplinary thinking [19]. In transdisciplinary thinking and research, "new connections among disciplines facilitate knowledge unification" [19]. For systems architects facing an uncertain future, historical parallels and past case studies may facilitate developing heuristics and specific techniques for a resilience strategy [15, 18]. Additionally, science fiction may help shape resilience strategy option sets via a tool such as the science fiction (SF) prototype, which is "a short story, movie or comic based specifically on a science fact for the purpose of exploring the implications, effects and ramifications of technology" [20]. Fusing this concept with the discipline of systems architecting and engineering enables systems thinking for unprecedented systems (such as the military AV SoS), which allows imagining and war-gaming various change cascades and their resultant disruptions on the system of interest (SOI). Also, such prototyping and imaginative simulation could reveal the expected scenarios for which the system would need broad utility. As Brian Johnson states, "The goal of SF prototyping is a conversation between science and the possibility of the future" [20]. Systems thinking expands this beyond the SOI to look at external factors, to include external disruptions. For example, an adjacent system in the SOI's expected operational environment may be limited by a lack of maturation on the Technological Readiness Level (TRL) scale [21]. As time progresses, how could that adjacent system behave as it moves from TRL 3 to 7, and how could that disrupt the SOI? Such a technique will not predict every disruption; however, it will help the systems architect explore the possible solutions for the system resilience strategy. Additionally, an SF Prototype may aid the stakeholders in refining their system objectives, requirements and options for the SOI [17, 20]. Upon implementation of a system, the SF Prototype products may serve as a baseline to develop more sophisticated training tools and model scenarios for human agents involved with the SOI.

6.2.3 Trade-Off Decisions

Trade-offs heavily influence the resilience strategy and architecting for resilience. The systems architect must conduct necessary trade-space analysis, which is a method "to analyze system architectures with respect to competing quality requirements," that helps "make informed architectural decisions" [22]. Architecting resilience into an AV SoS may increase cost and weight, which in turn could reduce other performance attributes. Analyzing trade-space effectively can allow stakeholders and "decision makers an understanding of capabilities, gaps, and potential compromises facilitating the achievement of system objectives" [23]. Looking at the current vehicle systems known as the high-mobility multipurpose wheeled vehicle (HMMWV) reveals a system that offers broad utility. This is evidenced by the numerous variants offered by AM General that serve different mission sets as well as the long life cycle for the system, ranging from 1984 to present day [11, 24, 25]. However, achieving broad utility came with a trade-off; the platform proved proficient in various uses over its life cycle, but the trade-offs made in the original platform impacted future unforeseen use cases. Asymmetric military operations in Iraq and Afghanistan revealed HMMWV vulnerabilities to IEDs, resulting in decreased survivability in the specific context of use [11, 26]. This led to significant investment in retrofitting HMMWVs with armor; in 2003, almost 20 years into the system life cycle, the DoD began investing "$1.2 billion for armored [HMMWVs] and armor kits" [27]. Interestingly, such retrofitting came with its own set of trade-offs. AM General's specifications list a weight gain of about 1600 pounds between the base platform and the M1151 up-armored HMMWV [24, 25]. Furthermore, studies via modeling and simulation concluded that the weight gain via added armor led to decreased braking performance while also increasing the "rollover propensity in the HMMWV" [28].

The preceding highlights the significance of trade-offs on resilience over the entire life cycle of a system. A system's architecture may have resilience when implemented in the expected contexts and timelines, but the trade-offs necessary to achieve resilience today may limit future resilience as the contexts evolve. Attempting to correct this mid-life cycle may force trade-off cascades. In the case of the HMMWV, using concepts from platform-based engineering (PBE) enabled the DoD's broad utility subcomponent of resilience but may have limited future system evolution, and thus future resilience as unknown contexts and use cases arose [29]. The platform bounded the system to "certain core assets, thereby limiting the useful life of the product family" [29]. Eventually, this led to the introduction of a new system known as the mine-resistant ambush-protected (MRAP) vehicle [30]. This is relevant to the AV SoS concept; current architectural trade-offs will have significant implications on resilience and system performance across the life cycle. For instance, the AMAS system is a kit that turns legacy soldier-driven vehicles into autonomous vehicles [31]. This prototype indicates trade-offs that value retaining legacy and current system interoperability over the costlier and more difficult proposition of a true clean sheet design that would take full advantage of automation. For a system as complex and unprecedented as the AV SoS, seemingly sound trade-offs may result in future trade-off cascades and fewer options as the complex system adapts and grows to meet shifting implementations, objectives, and contexts.

6.3 3. Introducing Resilience into the AV SoS

6.3.1 Systems Architecture and Resilience

Military AVs will replace some or all soldier-driven vehicles in military formations. There may be soldiers in some or all of the vehicles, but the extent to which they control the movement depends upon the trade-offs made in the systems architecture and its corresponding resilience strategy. This also impacts the ratio of autonomous to legacy systems in the formation. Given the resilience concepts, the following system attributes directly and indirectly contribute to resilience for this specific system: flexibility, adaptability, scalability, survivability, interoperability, and reconfigurability [22].

To achieve system resilience, AV SoS subsystems must work in concert so as to create emergence; this is a systems concept best surmised by Russell Ackoff, who said, "A system is not the sum of its parts but the product of the interactions of those parts" [8, 32]. This synergistic concept of the SoS depends upon the interfacing of various elements during system use, in which "[i]nteractions among entities leads to emergence" of new behaviors, functions, and outputs [8]. To achieve successful emergence, each AV must appropriately sense its environment and reference the information against a database, which may reside at a distant end accessible via communications links. Data sensors create a virtual representation of the world, able to be stored in libraries, analyzed, and referenced by other SoS members. While humans may be able to quickly identify objects via "rapid cognition" coupled with experience, the AV will have to use alternate methods [15]. The accuracy of the virtualized word depends upon sensor capability and numbers. Collected data can be stored locally or shared, leveraging multiple systems to process and converge overlapping data about a local virtualization. Humans may rapidly interpret their context due to intuition and inductive reasoning, but automated systems must rely on "deductive reasoning" as they interpret collected data and reference local or distant databases [33]. Eventually, AVs will gain rapid cognition of common scenarios encountered on their mission sets and use probabilistic analysis to determine which relevant data must be pulled forward for various mission sets.

The learning does not cease at a single vehicle; such acquisition of self and environmental knowledge can aggregate across the entire SoS, with other vehicles benefiting from continuously improving information. Such learning can also allow the system to absorb disruptions; for example, a specific AV's sensors may have failed. However, due to environmental information from other vehicles ahead and behind it with functioning sensors, the AV has enough awareness to successfully continue toward its objective. The preceding exemplifies the following resilience technique:

- *Learning and adaptation* – A resilient system is "continually acquiring new knowledge from the environment to reconfigure, reoptimize and grow" [15].

AV learning relies on a string of dependencies in a complex DoD SoS. First, is the AV capable of correctly visualizing its environment in a manner useful to the system and system adaptability [33]? Secondly, is there a persistent communication link with sufficient bandwidth to pass data to a repository that holds enough data to draw useful conclusions? Finally, is there enough computational power available to quickly formulate useful conclusions in a rapidly evolving, dynamic environment? The presence of such capabilities is critical for the achievement of the SoS's objectives and would further the AV SoS's adaptability and survivability by allowing it to successfully interrogate the environment and then avoid external disruptions. However, incorrectly architecting such a complex and complicated system could result in brittleness, a condition in which the system will "only achieve high performance under certain conditions" [15]. The ability of the AV SoS to reason depends upon the availability of the aforementioned capabilities. But the AV SoS will encounter complex environments and external disruptions, and a system as complex as a AV SoS will also encounter systemic disruptions. Thus, the following resilience technique captures techniques necessary to address issues such as stacked dependencies:

- *Functional redundancy* – "there should be alternative ways to perform a particular function that does not rely on the same physical systems" [15].

In the case of the adaptive learning, this may mean finding alternate methods to have learning occur. Instead of relying on distant data repositories and processing power, onboard systems may allow localized processing and data accrual, though the capabilities may not be as thorough as a dedicated processing center. Multiple AVs operating near each other may leverage concepts from distributed computing, in which "machines, interconnected with high-speed links perform different computationally intensive applications" [34]. Such a setup would utilize "mapping" for "[t]he matching of tasks to machines and scheduling the execution order" [34]. This is related to research in unmanned aerial vehicle (UAV) swarms by Edward Ordoukhanian, in which UAV smarms gain resilience through fluid transferal of critical operational tasks [35]. This is referred to as functional reallocation, in which the system will "[d]ynamically [redistribute] tasks among remaining UAVs upon the loss of [a] UAV (disruption)" [35]. These techniques yield resilience by preventing drift toward brittleness as external and systemic disruptions occur.

As an example, suppose four vehicles are traveling toward an objective and a disruption prevents access to the remote database and its processing capabilities. The lead vehicle focuses on pathfinding, the second vehicle on command and control. The third and fourth vehicles use distributed computing techniques to process the pathfinding calculation from the lead vehicle and build upon it by pooling sensory data from all four vehicles [34]. Expanding this beyond four vehicles, this four-vehicle element may have a global role of pathfinding for a vast network of AVs in an operating environment. In such an example, there exists a hierarchy among the AVs. Disruptions have the ability to impact that, and thus hierarchical fluidity becomes imperative. If a vehicle becomes disabled, hierarchies and functional allocations flow as necessary to other vehicles [35]. To work, though, this functional reallocation relies upon sufficient functional redundancy within the various AVs, which necessitates architectural decisions and appropriate trade-offs [35].

A flexible architecture enables such shifts in hierarchy or function, because persistent broadband communications are not given while operating in remote environments, and throughput availability is an expected change. Self-awareness of drift is also important, as a chain of disruptions may stack too many dependencies or functions upon two few nodes, resulting in brittleness [15]; "drift correction" combats this by allowing the system to "make appropriate trade-offs" while "taking timely preventive action," which may even include cessation of the operation to preserve the system [36].

6.3.2 Network Resilience

Adding network resilience and cyber resilience is critical to sustained operations for the system [15]. This resilience domain ensures persistent communication functionality and a cyberspace enclave continually usable by the AV SoS. Additionally, this resilience assures "smooth operation of distributed processing and networked storage" [15]. To be resilient in this domain, the network must adapt and learn about itself; as link availability fluctuates, a resilient network will actively reroute, prioritize, and optimize dataflow to maintain availability. It will also maintain awareness of data accessibility and distribute storage allocations as necessary. Conducting "real-time trade-offs" coupled with the ability to reason and adapt provides resilience while also creating the conduit for the desirable emergent behavior in the SoS [14].

There are dangers associated with the network; as in multi-UAV networks, use of "pre-determined communication paths can potentially work against the system" during normal use or during recovery "from a disruption" [35]. This leaves the system susceptible to brittleness; furthermore, a network that adapts to available links may become brittle as link availability diminishes due to disruptions. What few links exist may become saturated, inhibiting bandwidth and the data "flow rate" [35]. This may impact "response time" to critical information [35]. The ultimate effect of this compounding may be increased "recovery time" or even a complete inhibition of recovery due to other events and disruptions [35].

There are also challenges associated with a network that is too vast. Metcalfe's law posited that a network's importance or value "would increase quadratically" with the addition of new users [37]. The AV SoS's network may not follow Metcalfe's model perfectly, but the addition of more AVs (nodes) into the system increases the extent of the SoS's emergent behavior and potential value. This is due to having more AVs sensing and processing while also increasing more options for routing information. However, in an uncertain and contested environment, there are assurance and cybersecurity threats, and such a vast network may increase the cyber-attack surface of the SoS [38]. In this situation, more AVs means more nodes, interfaces, links, processors, and data storage, all of which represent entryways for malicious intrusion [38]. A high-value network comes with a large cyber-attack surface, which increases probability of intrusion as well as the leveragability of such an intrusion [37, 38]. Thus, Metcalfe's law applies to the cyber adversary, as a large network of high value to the system is also of high value to potential intruders.

Without the network, the AV SoS cannot exist, yet with the network comes rapidly scalable complexity and a host of new disruption possibilities. The artificial immune system (AIS) concept allows networked systems to mimic human immune systems and assist in network "intrusion detection" [39]. The AIS actively learns and adapts to recognize trusted and untrusted elements within a system and treat each accordingly, with untrusted or intrusive elements being recognized and treated as pathogens are treated in the human immune system [39]. Such a tool can allow the system to determine trusted AVs and control network membership; AVs that begin operating erratically or in a way so as to become considered a pathogen may have their network membership terminated and their data entries marked quarantined or, depending upon the contents, destroyed [39].

An incorrect application of the AIS and other network resilience tools may result in seeking resilience via added complexity; a systems architect may counter the problem of network complexity with elegant design [40]. A parsimoniously architected network will still achieve its objectives, maintaining "purposivity," while only accepting the "minimum number of components" necessary to do so [40]. The resultant architecture would adaptively bound the scalability of the network, with the network understanding when and how to scale and evolve so as to best maintain services in light of various disruptions [15, 40]. The system would also know when it lost too many nodes or connections necessary for mission accomplishment, allowing it to exercise "drift correction" and decide to abort its current objective in order to preserve the system [36]. Trade-offs weigh heavily on how such strategies are implemented, as stakeholders will have to determine network resilience's value relative to other resilience domains. The systems architect must look holistically at this trade-off and thoroughly consider alternatives with stakeholders, as the network critically integrates the various systems and is the pathway through which the desired emergence occurs.

6.3.3 Human-Systems Integration and Resilience

The degree to which soldiers are integrated into the control of such vehicles will greatly impact the resilience strategy. Initial prototypes as well as statements from the TARDEC director reinforce the idea that soldiers will have involvement in the control of the system. Dr. Rogers specifically stated, "Nothing can replace the life of a Soldier. Autonomy-enabled systems will help make the Army more expeditionary, keep Soldiers safe and make them more efficient" [8]. This shows a contrast with the traditional view in which human interaction with systems is viewed as a liability best minimized through design that segregates humans from machines wherever possible [9, 16, 33]. The challenge comes in making the appropriate functional delineation between automation-centric tasks and human-centric tasks, and it is a challenge that has impacted automated aircraft and spacecraft development [33, 41].

To assist in delineation, Jens Rasmussen created a behavioral taxonomy to determine task allocation [42]. Rasmussen's performance categories consist of "skill-based," "rule-based," and "knowledge-based" tasks [42]. In this hierarchy, automation is preferable for skill-based and rule-based tasks with lower uncertainty, both situations in which the responses are more methodical and memory-based [33]. Skill-based automation, however, is "highly dependent on the ability of [system] sensors to sense the environment and make adjustments" [33]. For rule-based tasks, "uncertainty management is key" [33]; this is because optimization grows less useful as uncertainty rises [22]. An example of this comes from a recent incident where Israeli soldiers relied on the Waze pathfinding application [43]. The application led them into hostile territory, which resulted in a skirmish [43]. According to the rules of the Waze application, the route was optimal, but the optimization rules failed to account for the militarily contested regions and their associated uncertainty [43]. Human intervention may have prevented the outcome by recognizing a need to refine the rules.

Humans may exceed automation's performance capacity for highly uncertain situations requiring knowledge and expertise, where "the human power of induction is critical to combat uncertainty" [33]. This alludes to the following resilience techniques:

- *"[H]umans should be able to back up automation when there is a context change that automation is not sensitive to and when there is sufficient time for human intervention"* [15].
- *"[H]umans should be in the loop when there is a need for 'rapid cognition' and creative option generation"* [15].

To illustrate the preceding, consider a military convoy moving across rural terrain. The lead vehicle encounters a herd of cattle and a herder walking down the opposing traffic lane. The simple act of driving may be a skill-based event based on "sensory-motor actions that are highly automatic," thus being suitable for automation [33]. Furthermore, it may be bound by the rules of the road or rule settings for basic "path planning" [33]. However, the presence of cattle and the herder create the possibility of an external disruption via an accident. For automation to be successful, "uncertainty should be low and sensor reliability high" [33]. Assuming perfect sensors, the AV SoS can produce an accurate image, to include data well beyond human capabilities such as object velocity, distance, location coordinates, object counts, and so on. However, the advantage the human observer has is rapid cognition and the ability to apply knowledge and experience in order to interpret the context [33, 36]. The human observer cannot determine the exact number of cattle, their velocities, or their precise coordinates; however, a human observer can rapidly determine that the cattle are in a herd guided by another human and the herd can be reasonably expected to move as a predictable unit. This is inductive reasoning; conversely, automated sensors would have to use deductive techniques in an attempt to match sensed information to similar datasets in whatever database is available [33]. This example highlights how system resilience necessitates that the AV SoS and soldiers work in concert to leverage their unique skills. Ungraceful integration may turn otherwise negotiable situations into sources of brittleness, while successful integration can result in resilience and positive SoS emergent behavior.

To further the emergence and thus resilience, the soldier in the vehicle must understand what the AV's intentions are as it reasons about an uncertain or complex situation. This requires the soldier to retain contextual awareness while the system must defer the "final say" to the soldier [16]. It may also allow for additional system learning; for example, the soldier may be aware of strategic implications such as tense military-civil relations, requiring a wider and slower path around the cattle and herder in the previous example. In a harmoniously synched system, the soldier and AV would jointly apply and share sensory, automated knowledge with human knowledge and expertise during decision cycles [33]. This will contribute to resilience by allowing the repelling and avoidance of external disruptions or, if disrupted, by making the best follow-on decisions given the rapidly evolving and uncertain environment. This is embodied by the resilience concept of "intent awareness," in which the human and the system "maintain a shared intent model to back up each other when called upon" [15].

To ensure such an emergent behavior between human and system, a sound interface is imperative. When architecting, "The greatest leverage in system architecting is at the interface" [44]. As AVs move beyond retrofit kits of human-driven vehicles such as AMAS [31], the design of purely autonomous vehicles will shift in a way similar to how aircraft have "digital fly-by-wire" and autopilot systems [33]. Traditional analog vehicle inputs may give way to repackaged vehicles that have radically redefined touchscreen control interfaces such as Arturo Reuschenbach, Miao Wang, Tinosch Ganjineh, and Daniel Gohring's prototype "iDriver remote software on [the] Apple iPad" [45]. Such a redefined control system allows for a familiar touchscreen interface that can implement elegant design, a method for shielding users from inherent complexity so they can effectively and easily use a system [40]. The challenge for the architect is to ensure such an interface displays appropriate functions and information when needed.

Not only is the right information delivery imperative, but it is also imperative that soldiers are able to understand when sensors and interfaces may be experiencing systemic disruptions. "Automation- induced complacency" is a potential systemic disruption in which the soldier may rely on incorrect or even faulty AV sensory data and decisions instead of "their own vigilant information seeking and cognitive processing" [9]. Hence, overreliance on automation could lead to a situation in which soldiers default to the automation's decision, even if it is wrong or malfunctioning. The systems architect's ability to correctly place the human role within the system is imperative to avoiding, repelling, and managing disruptions [9].

Discussing system trust alludes to two additional resilience techniques applicable to human-systems integration:

- *Predictability* – "automated systems should behave in predictable ways to assure trust and not evoke frequent human over-ride" [15].
- *Reorganization* – a "system should be able to restructure itself in response to external change" [15].

In war, soldiers routinely encounter uncertainty. To prepare, units train in basic individual skills related to the concepts of "shoot, move, and communicate" while also training in collective skills such as battle drills [46, 47]. Battle drills entail concepts "such as movement to contact" that integrate individual skills into collective action [48]. These drills condition groups to expect certain behaviors, resulting in "synchronized... lines of action among individuals and groups" [48]. The result is that in the disjointed chaos of war, soldiers react with predictable techniques and battle drills that begin to establish some degree of order, which in turn creates future possibilities and options for achieving an objective. Mission planning complements this with sound commander's intent. This is a military concept that communicates the desired, high-level outcome of an operation without all of the specific details so that in the event of disruptions to the mission and leadership, "the lowest-ranking person is still able to carry out the mission" [49]. General Patton's famous line, "Never tell people how to do things. Tell them what to do and they will surprise you with their ingenuity," captures this concept [50]. The what to do in General Patton's quote is commander's intent, with the how being encapsulated in the details of mission planning or the how comes from ad hoc ingenuity in the face of unimaginable uncertainty [49, 50].

Injecting AVs adds autonomous agents into the military formation; such agents in a well-architected system will exhibit adaptability, to include "anticipation, responding, [and] learning," as well as the ability to make dynamic decisions [14]. The AVs must respond and act in a manner that fosters trust and predictability lest they risk losing the confidence of their human counterparts [15]. As the AV SoS reorganizes through learning, it must do so in a way that complements the behaviors and expectations of soldiers while not exceeding the human "adaptation rate," which is the "upper bound on how fast systems can adapt" [9]. This means the AV SoS may have to occasionally delay its adaptation and decision making in certain situations so as to not exceed the adaptation rate, decision cycle rate, and processing speed of soldiers. Though it sounds counterintuitive, this may yield higher performance as such a scenario would enable predictability and behavior that leverages the joint capabilities of automation and human processing, the interactivity of which has a greater performance potential for a complex AV SoS than that of a system optimized in favor of pure technological capabilities [9, 16].

Such synching of human and automation relies heavily on architectural decisions, especially at the critical interfaces. Once established, though, the system could learn and adapt to the soldiers it works with in training, jointly conducting battle drills and other scenarios [47, 48]. Self-awareness and reasoning about its system state includes awareness of soldiers and learning from their behaviors.

Complementary learning between soldier and machine will enable the AV SoS and its operators to actively exemplify General Patton's maxim on ingenuity in the face of uncertainty, as the system will move beyond default, programmed responses to mutually learned responses [50]. The final key to such trust and cohesiveness, though, is proper architecting of the aforementioned "final say" [16]. While history is replete with examples of soldiers making great sacrifices in battle, an AV SoS that makes such a decision on behalf of the soldier would greatly endanger the necessary trust between soldier and system [16]. Engendering the "bond of confidence" means correctly determining the "circumstances when the human should not be allowed to override the system" and vice versa [16]. As such, a decision from the AV that knowingly endangers the life of the soldier must be made in consensus with the soldier while granting that soldier final say prior to execution [16]. To architect the system otherwise would deplete confidence and likely result in rejection of the system.

6.4 Conclusion

While several techniques for resilience have been offered in this paper, the true challenge is in selectively integrating different techniques that are pertinent for the SOI. In the discussed resilience subdomains of network, cyber, and human-systems integration resilience, all combine to form the AV SoS's resilience. However, trade-offs play a significant role in a system's overall resilience, and specific resilience attributes, domains, and techniques must be weighed against each other [51]. Hidden interdependencies result in counterintuitive impacts on resilience; as a system's learning capacity and thus the requisite components are scaled up, it may adversely impact network and cyber resilience by increasing cyber-attack surface [38, 51]. Increasing network and cyber resilience may create a highly secure system that impacts the ability of the user to interact with it or does not provide ample flexibility and adaptability for the overall systems architecture. Legacy integration may introduce brittle nodes in an otherwise resilient system. Additionally, resilience is not free; adding resilience may impact affordability, resulting in a system vulnerable to external economic and political volatility [51, 52]. In such a scenario, an otherwise resilient system may not have the economic support system necessary to maintain it. Finally, in light of the time value of money and risk analyses, initial investment in resilience may be a worse financial alternative to simply replacing the system in the future [51, 52].

Determining how to architect for SoS resilience and how to balance the interplay of various techniques is complex, but the aforementioned systems thinking, SF prototyping, and transdisciplinary thinking may provide some clarity [18, 20, 53]. Additionally, experiential design, in which "storytelling in virtual worlds" couples with model-based systems engineering (MBSE), can reveal how individuals "interact with the system" and how various environments, events, and event sequences interplay with the system [54]. Such techniques filter out the true system objectives from stakeholders while allowing a reasonable scoping of potential use cases and system scenarios, which in turn drives the resilience strategy [54]. Use of modeling and simulation throughout will enable trade-space analysis and assist in revealing counterintuitive impacts from scaling various resilience attributes [51, 54]. This relates to the importance of systems thinking and the architecting process, as understanding the needs and objectives in light of the holistic system across the course of its life cycle directly impacts the meaningfulness and relevance of a resilience strategy. Sound systems architecture enables successful system design [8]; similarly, sound systems architecting enables a useful resilience strategy, which in turn enables a resilient design. The front end architecting phase sets the constraints on where resilience can be introduced. Thus, it is important to ensure that the systems architecture is flexible in the right dimensions to introduce resilience mechanisms and solutions.

References

1. Office of the Deputy Under Secretary of Defense for Acquisition and Technology, Systems and Software Engineering (2008) Systems engineering guide for systems of systems, 1st edn. DoD, Washington, DC
2. Art Money and Bill Hilsman (2009) Creating an assured joint DoD and interagency interoperable net-centric enterprise. DoD, Washington, DC
3. Lockheed Martin (2014) US Army and Lockheed Martin complete second series of advanced autonomous convoy demonstrations [Online]. Available: http://www.lockheedmartin.com/us/news/press-releases/2014/june/mfc-061014-us-amy-lockheedmartin-complete-second-series-advanced-autonomous-convoy-demonstrations.html. Accessed 14 Feb 2016
4. *TARDEC 30-YEAR STRATEGY*, Version 1.1, TARDEC, Warren, MI, 2014
5. The Official Home Page of the United States Army (2014) Leading *Army researcher: Future of autonomous vehicles* [Online]. Available: http://www.army.mil/article/139889/LeadingArmyresearcherFutureofautonomousevehicles/. Accessed 13 Feb 2016
6. Bornstein J, Mitchell B (2012) Robotics collaborative technology alliance – Foundations of autonomy for ground robotics, Army Resarch Lab, Adelphi. Available at: http://www.arl.army.mil/www/pages/392/OverviewBriefingRoboticsCTA.pdf. Accessed 14 Feb 2015
7. Bruno R, Conti M, Gregori E (2005) Mesh networks: commodity multihop ad hoc networks. IEEE Commun Mag 43(3):123–131
8. Madni AM (unpublished) Systems architecture: key aspect of resilience/Spring 2016 – Lecture 4
9. Madni AM (2011) Integrating humans with and within complex systems. CrossTalk 24(3):4–8
10. Cohen RA (2011) Yerkes-Dodson law. In: Encyclopedia of clinical neuropsychology. Springer, New York, pp 2737–2738
11. Georger SR, Madni AM, Eslinger OJ (2014) Engineered resilient systems: a DoD perspective. Procedia Comput Sci 28:865–872
12. Wadud Z, Marsden G (2016) Self-driving cars/will they reduce energy use?, University of Leeds Centre for Integrated Energy Research, Leeds
13. Neches R, Madni AM (2013) Towards affordably adaptable and effective systems. Syst Eng 16 (2):224–234
14. Madni AM (unpublished) Overview/Spring 2016 – Lecture 1
15. Madni AM, Jackson S (2009) Towards a conceptual framework for resilience engineering. IEEE Syst J 3(2):181–191

16. Madni AM (2010) Integrating humans with software systems: technical challenges and a research agenda. Syst Eng 13(3):232–245
17. Madni AM (2013) Generating novel options during systems architecting: psychological principles, systems thinking, and computer-based aiding. Syst Eng 17(1):1–9
18. Madni AM (unpublished) Systems thinking/Courtesy of: intelligent systems technology inc./ Spring 2016 – Lecture 3
19. Madni AM (2007) Transdisciplinarity: reaching beyond disciplines to find connections. J Integr Des Process Sci 11(1):1–3
20. Johnson BD (2011) Science fiction prototyping: designing the future with science fiction. Synth Lect Comput Sci 3(1):1–190
21. NASA (2007) Systems engineering handbook, 1st edn. NASA, Hanover
22. Madni AM (unpublished) Tradespace analysis in engineered resilient systems and sos/Spring 2016 – Lecture 5
23. Spero E, Avera MP, Valdez PE, Georger SR (2014) Tradespace exploration for the engineering of resilient systems. Procedia Comput Sci 28:591–600
24. AM General (2016) Spec Sheets I HMMWV (Humvee) [Online]. Available: http://www.amgeneral.com/vehicles/hmmwv/specs.php. Accessed 20 Mar 2016
25. AM General (2016) Vehicles I A2 Series I HMMWV (Humvee) [Online]. Available: http://www.amgeneral.com/vehicles/hmmwv/a2-series/vehicles. Accessed 20 Mar 2016
26. Cameron RS (2007) "Scouts out – but not in hmmwvs!," *Armor*, pp. 26–32. Available: http://www.benning.army.mil/armor/historian/content/PDF/HMMWV%20Scouts.pdf
27. U.S. Department of Defense (2004) Armored Humvees, Tactics Address IED Threats [Online]. Available: http://archive.defense.gov/news/newsarticle.aspx?id=24640. Accessed 8 Mar 2016
28. Mica G, Marvi H, Arakere G, Bell WC, Haque I (2010) The effect of up-armoring of the high-mobility multi-purpose wheeled vehicle (HMMWV) on the off-road vehicle performance. Multidiscip Model Mater Struct 6(2):229–256
29. Madni AM (2012) Adaptable platform-based engineering: key enablers and outlook for the future. Syst Eng 15(1):95–107
30. U.S. Department of Defense (2007) MRAP (Mine Resistant Ambush Protected) vehicles [Online]. Available: http://archive.defense.gov/home/features/2007/mrap/. Accessed 20 Mar 2016
31. Lockheed Martin (2016) Autonomous Mobility Applique System (AMAS) [Online]. Available: http://www.lockheedmartin.com/us/products/amas1.html. Accessed 14 Feb 2016
32. McDevitt CA, Cahill EC, Stambaugh C (2004) System- level application of the evolutionary product development process to manufactured goods. Syst Eng 7(2):144–152
33. Cummings MM (2014) Man versus machine or man+ machine? IEEE Intell Syst 29(5):62–69
34. Braun TD et al (2001) A comparison of eleven static heuristics for mapping a class of independent tasks onto heterogeneous distributed computing systems. J Parallel Distrib Comput 61(6):810–837
35. Ordoukhanian E, Madni MA (2016) Resilient multi-uav operation: key concepts and challenges, 54th AIAA Aerospace Sciences Meeting
36. Madni AM (unpublished) Resilience concepts and current limitation/Spring 2016 – Lecture 2
37. Briscoe B, Tilly B (2006) Metcalfe's law is wrong- communications networks increase in value as they add members-but by how much? IEEE Spectr 43(7):34–39
38. Hahn A, Govindarasu M (2011) Cyber attack exposure evaluation framework for the smart grid. IEEE Trans Smart Grid 2(4):835–843
39. Le Bodec J-Y, Sarafijanovic S (2004) An artificial immune system approach to misbehavior detection in mobile ad hoc networks. In: Biologically inspired approaches to advanced information technology. Springer, Berlin, pp 396–411
40. Madni AM (2012) Elegant systems design: creative fusion of simplicity and power. Syst Eng 15(3):347–354

41. Liang S, Cummings ML, Smith CA (2008) Past, present and future implications of human supervisory control in space missions. Acta Astronaut 62(10):648–655
42. Rasmussen J (1983) Skills, rules, and knowledge; signals, signs and symbols, and other distinctions in human performance models. IEEE Trans Syst Man Cybern 3:257–266
43. The Washington Post (Mar. 1, 2016). *Israeli troops relying on Waze app blunder into Palestinian area; clashes follow* [Online]. Available: https://www.washingtonpost.com/world/middleeast/battles-erupts- after-israeli-soldiers-follow-apparent-gps-error-into-palestinian-zone/2016/03/01/940307ef-503f-4a98-8abb-01cf6357a850 story.html. Accessed 8 Mar 2016
44. Rechtin E, Architecting S (1991) Creating and building complex systems. Prentice Hall, Englewood Cliffs, p 312
45. Reuschenbach A, Wang M, Ganjineh T, Gohring D (2011) idriver-human machine interface for autonomous cars, 8th international conferences on information technology: new generations (ITNG), IEEE
46. The Official Homepage of the United States Army (2015) Shoot, move, and communicate! [Online]. Available: http://www.army.mil/article/147332/Shoot move and communicate/. Accessed 2 Apr 2016
47. Army Posture Statement 2008 (2008) 2008 Army posture statement I warrior tasks and battle drills [Online]. Available: http://www.army.mil/aps/08/information papers/prepare/Warrior Tasks and Battle Drills.html. Accessed 2 Apr 2016
48. Siebold GL (2007) The essence of military group cohesion. Armed Forces Soc 33(2):286–295
49. Augustine S, Payne B, Sencindiver F, Woodcock S (2005) Agile project management: steering from the edges. Commun ACM 48(12):85–89
50. Souba WW (2001) Leadership and strategic alignment-getting people on board and engaged. J Surg Res 96(2):144–151
51. Wheaton MJ, Madni AM (2015) Resiliency and affordability attributes in a system integration tradespace, AIAA Space 20J5 Conference and Exposition, pp 1–9
52. Wheaton MJ (unpublished) Affordable resilient systems/Guest lecture sae 599
53. Madni AM (2017) Transdisciplinary systems engineering: exploiting convergence in a hyper-connected world. New York, Springer
54. Madni AM, Nance M, Richey M, Hubbard W, Hanneman L (2014) Toward an experiential design language: augmenting model-based systems engineering with technical storytelling in virtual worlds. Procedia Comput Sci 28:848–856

Chapter 7
A Robust Portfolio Optimization Approach Using Parametric Piecewise Linear Models of System Dependencies

Navindran Davendralingam, Cesare Guariniello, and Daniel Delaurentis

Abstract "System of Systems" architecture problems can be very challenging due to the large number of systems involved and complex interdependencies that exist between and among them. The high number of decision variables and interactions presents the need for an appropriate collection of methods, processes, and tools that can help practitioners deal with such complexities and answer key questions that typically arise when evolving an architecture. In this chapter, we build on prior work toward developing a System-of-Systems Analytic Workbench and propose a combined use of two of the workbench tools, namely, the Robust Portfolio Optimization and Systems Operational Dependency Analysis tools. The purpose of the combination is to more explicitly introduce the dependency modeling capabilities of one with the computationally efficient decision-support capabilities of the other. We demonstrate application of the proposed combined approach on a conceptual Naval Warfare Scenario problem.

Keywords Systems portfolio • Architecture • Dependencies • System-of-systems

7.1 Introduction

System-of-systems (SoS) engineering and architecting is difficult due to a range of factors that includes the large number of systems involved, and the complex interdependencies that exist among and between the constituent systems that comprise the overall SoS. In a SoS, each component system typically operates independently and is managed at least in part for its own purpose [2]. The nature of dependencies between constituent systems directly impacts the behavior and functionality of the SoS as a whole [1]. Since the elements are designed and developed independently, the aggregate behavior emerges through the interactions between the elements, and even can produce unexpected, emergent behavior [3, 4]. In

N. Davendralingam (✉) • C. Guariniello • D. Delaurentis
Purdue University, School of Aeronautics and Astronautics, West Lafayette, IN, USA
e-mail: davendra@purdue.edu; cguarini@purdue.edu; ddelaure@purdue.edu

A.M. Madni et al. (eds.), *Disciplinary Convergence in Systems Engineering Research*, https://doi.org/10.1007/978-3-319-62217-0_7

addition, the behavior of a SoS is strongly evolutionary, since not only each individual component system has its own dynamics, but also the interactions between the systems can change over time.

Another difficulty concerning the modeling and analysis of systems-of-systems arises from the variety of possible aspects in the life cycle of a SoS that can be of interest to practitioners and users. De Weck et al. [6] investigated the semantic relationships between many these lifecycle properties. Their study underlines the multifaceted nature of SoS problems. Based on similar considerations, Davendralingam et al. proposed and developed an analytic work bench for SoS, featuring various tools and underlying methodologies, each one addressing a specific aspect of SoS modeling and analysis [7]. The System-of-Systems Analytical Workbench (SoS-AWB) [7] is a part of research efforts funded under the auspices of the Department of Defense (DoD)'s Systems Engineering Research Center (SERC), to address some of the key issues that arise when evolving a complex systems architecture. Current guidelines in the DoD Defense Acquisition Guidebook (DAG) (5000 series), and the SoSE guide, provide only high-level guidance in dealing with systems-of-system problems, prompting the need for more rigorous methods, processes, and tools to aid SoS practitioners in evolving DoD architectures.

In this work, we present a combined use of two of the set of methodologies available in the SoS-AWB, namely, the Robust Portfolio Optimization (RPO) [8] and Systems Operational Dependency Analysis (SODA) [9] methods. The purpose of this union of methods is to leverage the interdependency modeling benefits of SODA within the robust optimization-based tradespace exploration framework of RPO. SODA allows for complex interactions between individual systems to be captured as a series of intuitively parameterized piecewise-linear functions. RPO on the other hand models systems as 'building blocks' (with specific behaviors) and leverages innovations in robust optimization techniques to address architectural decision-making and tradespace exploration, under various data uncertainty. Jointly, the methods allow for (1) inclusion of improved information on connectivity behaviors (via SODA) within the decision-making construct of RPO and (2) formation of the joint framework in a manner that leverages the benefits in computational efficiency of mixed-integer programming (MIP) techniques. These benefits enable the analysis of complex system interactions in early-stage SoS tradespace and analysis in a manner that can also account for a range of uncertainties that are inherent in early-stage SoS evolution.

A brief review of the concept of analytic workbench and of RPO and SODA is described in Sect. 7.2. Section 7.3 describes the motivation behind the choice of these two tools for this work, the advantages of the combined use of the tools, and the formulation of the combined problem. In Sect. 7.4, we present the results of a small case study, to illustrate the combined application of RPO and SODA. Conclusions are drawn in Sect. 7.5.

7.2 An Analytic Work Bench Perspective

The SoS-AWB generally address issues related to system and connectivity selection, cost, performance, schedule, and risk for a given SoS architecture, by addressing some of the key common questions that arise throughout the evolution lifecycle [7]. (We invite the reader to reference [7] for a more comprehensive introduction to the SoS-AWB and range of methods within it). Some of the questions to be addressed include (but not limited to) *How to assess consequences of changes in architecture? Where, what, and how do risks change with changes in system parameters? How and where to effect change for risk mitigation? What systems (or connectivities) should be added or removed?* While the tools in the workbench address technical and programmatic complexities, they remain domain agnostic and therefore suitable to be applied to a variety of SoS problems.

7.2.1 Robust Portfolio Optimization

A systems-of-systems modeling and analysis problem can be sometimes described as a combinatorial problem to determine the most promising portfolio of individual systems to be invested on to achieve a certain capability. For instance, in warfare system-of-systems, this notion indicates that the systems architect needs to find the best combination of equipment, weaponry, facilities, use of existing resources, and other capabilities to achieve the required goal in various warfare scenarios. The process of choosing the "best" portfolio should also take account of cost and uncertainty factors along with the desired capability. In this context, RPO methodologies, initially developed for financial investment problems, can support the practitioners and decision makers.

The purpose of the RPO methodology is to maximize the expected system-of-systems performance that is subject to constraints such as connectivity, risk, and cost, while utilizing innovations from the area of robust optimization to address various forms of data uncertainty. Thus, implementation of RPO for a certain system-of-systems design problem yields a set of Pareto optimal portfolios corresponding to a user-defined risk aversion factor. The optimization is based on a mixed-integer programming technique and all the interdependency between component systems are depicted as constraints (Fig. 7.1).

In the RPO methodology, a system-of-systems is modeled as an interconnected set of nodes, each with a predefined set of inputs and outputs. The inputs represent the requirements of the node, and the outputs represent the capabilities of the node. The set of constraints and rules on inputs and outputs allow for the development of feasible architectures. The overall performance of a system-of-systems is based on the ability of the connected network of systems to fulfill the overarching desired objectives.

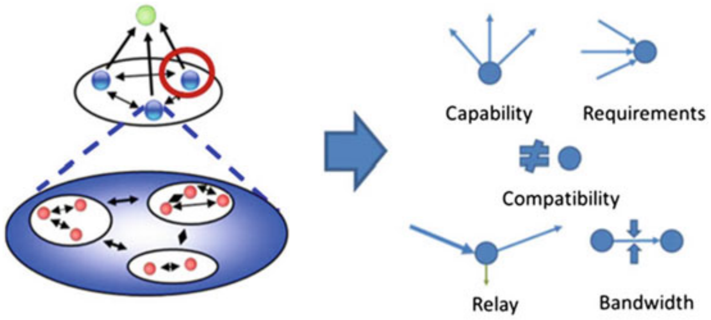


Fig. 7.1 Various common forms of dependencies between systems are modeled and treated as constraints

7.2.2 Systems Operational Dependency Analysis Tool

The Dependency Analysis Methodology, based on the concepts of Functional Dependency Network Analysis [10, 11], has the goal of addressing some of the limitations of traditional systems engineering approach when dealing with complex systems. This methodology assesses the effect of dependencies among components in a monolithic complex system, or among systems in a system-of-systems, in the operational domain [12, 13].

SODA methodology is based on a parametric model of the behavior of the system [9]. In SODA, the system-of-systems is models as a dependency network, where nodes represent the constituent systems and the capabilities that the SoS is required to achieve. The edges represent the operational or developmental dependencies. Each node is characterized by a measure of self-effectiveness (SE), which quantifies its internal health status. Each link is characterized by three parameters: α, β, and γ quantify, respectively, the Strength of Dependency (SOD), Criticality of Dependency (COD), and Impact of Dependency (IOD) that affect the behavior of the whole system-of-systems in different ways. Figure 7.2 shows a small SODA dependency network, and Fig. 7.3 shows how the parameters affect the input/output model between two nodes. The representation of a system-of-systems as a network prevents the method from being domain-dependent and allows for its application across various classes of problems.

This approach has multiple advantages:

- It results in a convenient model, with parameters that have an intuitive meaning, directly related to the features of the dependency. Therefore, they give a direct insight into the causes of observed, and possibly emergent, behavior.
- It supports informed decision making in design and update of systems and system-of-systems architecture, reducing the amount of interrogative operations, such as simulation, required to obtain the necessary information.

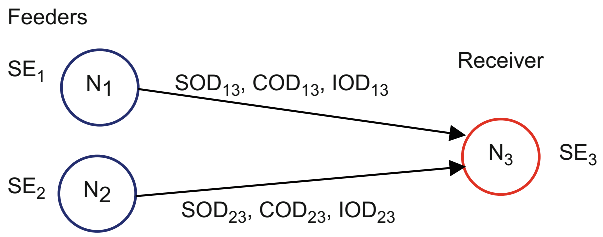

Feeders
SE$_1$
N$_1$
SE$_2$
N$_2$
SOD$_{13}$, COD$_{13}$, IOD$_{13}$
SOD$_{23}$, COD$_{23}$, IOD$_{23}$
Receiver
N$_3$
SE$_3$

Fig. 7.2 Synthetic SODA network. *N* node, *SOD* strength of dependency, *COD* criticality of dependency, *IOD* impact of dependency, and *SE* self-effectiveness

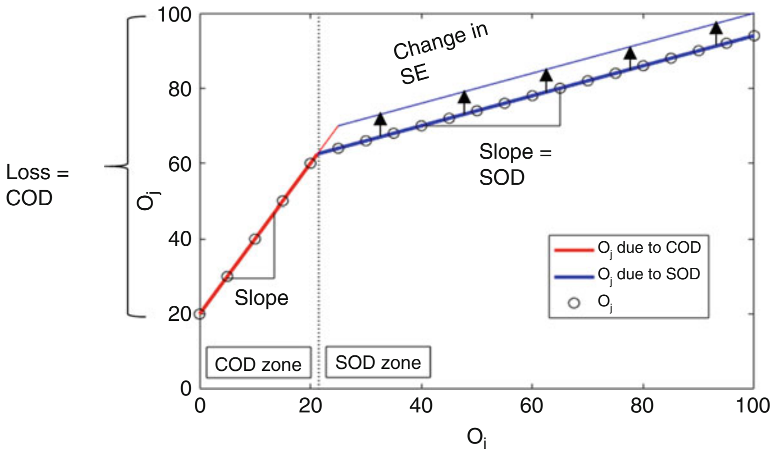


Fig. 7.3 Operability *Oj* of a dependent node *j* in function of the operability *Oi* of a feeder node *i*. *SE* self-effectiveness, *SOD* strength of dependency, *COD* criticality of dependency, *IOD* impact of dependency

- The parameters of the model can be quantitatively linked to a range of possible input sources, including (but not limited to) experiments, historical data, and subject matter expert evaluation.

The goal of this methodology is to provide a framework to support decision in systems design, development, and architecture. Using Dependency Analysis methodology, designers and decision makers can quickly analyze and explore the behavior of complex systems in the operational domain, and evaluate different architectures under various working conditions. Based on the raw results of the tools, the user can quantify various metrics of interest (e.g., robustness, resilience, and delay absorption), compare architectures based on these metrics, perform trade-off between competing desired features, ascertain criticalities, identify the most promising architectures, and discard architectures the lack the requested features. This way, in early design process, promising architectures can be kept into consideration and improved based on the information given by the model parameters and the observed behavior, thus supporting the process of concept selection.

7.3 Combined Use of Robust Portfolio Optimization Methodology and Systems Operational Dependency Analysis Methodology

7.3.1 Motivation for the Choice of Tools and Advantages of Their Combined Use

The main goal of SODA is to analyze and assess properties of a whole architecture at the SoS level, for example, robustness and resilience, and the impact of various operational disruptions and uncertainty on these properties. However, each instance of SODA analysis operates on a given architecture, made of predetermined "nodes," and various levels of uncertainty on the status and capabilities of each system are provided as input to SODA. In short, SODA provides a quantification of *relational* dependencies that exist between nodes. Choices made on a system to fulfill the role of a node may affect the nodes ability to "feed" a certain degree of operability but does not change the relational information of the SODA model.

RPO on the other hand addresses the choice of selecting collections of available systems to construct a 'portfolio' of systems that are subject to constraints on how the constituent systems interact, and with consideration for various forms of data uncertainty. RPO is posed in terms of a mixed-integer program and assumes linear additive contributions of nodes to an assumed performance index. However, the relational information of SODA can be introduced, through the addition of auxiliary variables as a set of linear constraints within RPO, preserving the mixed-integer programming structure of RPO.

The combined use of SODA and RPO is envisioned to begin with the SODA analysis yielding the deterministic relational components of nodes within the SoS. Individual systems can be used to act as feeder nodes within the SODA network; this naturally limits the analysis to the case of SoS architectures where the relational components are assumed to be static and that the feeder nodes are of specific interest. In the joined application of the two tools, RPO provides architectural choices to SODA, selecting from an available portfolio of systems at each feeder node, with each system having its own properties in terms of capabilities and uncertainties. Therefore, the combined use of the two allows for the selection of an optimal portfolio of systems (under data uncertainty and preserving chosen degree of robustness) at the feeder node level, with the objective function of the RPO reflecting the performance index of capabilities that propagate as a result of the piecewise linear relationships across the relational nodes.

7.3.2 Problem Formulation

The joint RPO-SODA optimization problem is posed as a mathematical programming problem where the objective is to maximize some expected operability of the network, while satisfying various connectivity and resource constraints imposed on the candidate systems to be selected. The resulting equations can be written as the following:

$$\max\left\{\mathrm{O}_{\mathrm{i=target}}\right\} \tag{7.1}$$

s.t.

Dependency Constraints

$$\mathrm{o}_j = \left\{\begin{matrix} f_{\mathrm{COD}}(\mathrm{o}_i) \text{ if } \mathrm{o}_i \le \delta_i \\ f_{\mathrm{SOD}}(\mathrm{o}_i)\text{if } \mathrm{o}_i \ge \delta_i \end{matrix}\right\}\text{(piece-wise linear − single dependencies)}$$

$$\tag{7.2a, b}$$

$$\mathrm{o}_j = \frac{1}{n}\sum_{i=1}^{n} \mathrm{o}_i\text{(for multiple dependencies)}$$

Connectivity and Resource Constraints at Feeder Nodes

$$\sum_{j} x_{\mathrm{cij}} \le x_i^B S_{\mathrm{ci}} \tag{7.3}$$

$$\sum_{i} x_{\mathrm{cij}} \ge x_j^B S_{\mathrm{rj}} \tag{7.4}$$

$$x_1 + \cdots + x_n = \mathrm{L} \tag{7.5}$$

$$\sum_{c} x_{\mathrm{cij}} - x_{\mathrm{ij}} M \le 0 \tag{7.6}$$

$$M\sum_{c} x_{cij} - x_{ij} \ge 0 \tag{7.7}$$

$$x_{\mathrm{ij}} \le \mathrm{Limit}_{\mathrm{ij}} \tag{7.8}$$

$$\sum_{i} x_{\mathrm{cij}} - \sum_{j} x_{\mathrm{cij}} - x_j^B S_{\mathrm{rj}} = 0 \tag{7.9}$$

$$x_{\mathrm{cij}} = 0 \quad c \in \text{capability} \tag{7.10}$$

$$x_{\mathrm{cij}} \in \text{real, binary}, x_j^B \in \text{binary}$$

Equation (7.1), the objective function, seeks to maximize the operability of the target end node (here, overall operability). Equation (7.2a, b) describes the dependencies of operability between constituent nodes on the SODA network. Single dependencies between two nodes are quantified in terms of piecewise linear functions (based on SODA representation). In the case of multiple dependencies that stem from multiple feeder nodes, the operability of the receiver node is given as the average of operability from feeders. Equations (7.3, 7.4, 7.5, 7.6, 7.7, 7.8, and 7.9) follow the same formulation of constraints as previously modeled in [8] and reflect the various types of connectivities and flow of resources as shown in Fig. 7.1. Note that in the context of this joint SODA-RPO work, the connectivities reflect the combination of systems that given rise to the operability at each feeder node locally, and not the operational SODA network. While Eq. (7.2a) is piecewise linear, there are standard conversions of such constraints, using auxiliary variables, into linear constraint – this makes the overall problem a mixed integer programming (MIP) problem.

7.3.3 Robust Portfolio Optimization: Using Bertsimas-Sim Approach

The preceding equations denote the deterministic description of our architecture problem. However, given the uncertainty that can manifest in the model, with focus on the feeder nodes of the network, we leverage recent advances in the area of robust optimization to allow for decision-making and optimization under conditions of bounded uncertainty. In this chapter, and in our prior works, we have adopted the approach from Bertsimas and Sim's seminal work in [14] to deal with uncertainty in linear constraints. The Bertsimas-Sim method [14] is a robust linear formulation that addresses parametric data uncertainty without excessively penalizing the objective function [14]. The method affords the user the means of building in a specific amount of robustness, through manipulation of constraint-wise parameters, and is naturally extendable to discrete optimization problems – a very attractive feature in the case of selecting discrete systems for an architecture.

General inequality constraints in linear programming problem are normally written as Ax $\leq$ b. Here, a subset of matrix A_{ij} is assumed to have uncertainties (data-driven uncertainties) that exist within symmetric intervals, that is, belong to set J_{i} and $\left[a_{\mathrm{ij}} - \hat{a}_{\mathrm{ij}}, a_{\mathrm{ij}} + \hat{a}_{\mathrm{ij}}\right]$. A conservatism parameter, Γ_{i}, is used to adjust the degree of conservatism in protection of the uncertain linear constraints from infeasibility; Γ_{i} takes values in the interval [0, $|J_{\mathrm{i}}|$] where J_{i} represents the set of A_{ij} coefficients that are uncertain and is not necessarily integer. The proof, as provided in literature, forms the problem into the following linear optimization problem:

$$\text{maximize} \qquad c^{\mathrm{T}}x \tag{7.11}$$

subject to:

$$\sum_{j} A_{\mathrm{ij}} x_{\mathrm{j}} + z_{\mathrm{i}} \Gamma_{\mathrm{i}} + \sum_{j \in J_{\mathrm{i}}} p_{\mathrm{ij}} \leq b_{\mathrm{i}} \tag{7.12}$$

$$z_{\mathrm{i}} + p_{\mathrm{ij}} \geq \hat{a}_{\mathrm{ij}} y_{\mathrm{j}} \tag{7.13}$$

$$-y_j \leq x_j \leq y_j \tag{7.14}$$

$$l_j \leq x_j \leq y_j \tag{7.15}$$

$$p_{\mathrm{ij}}, y_{\mathrm{ij}}, z_{\mathrm{ij}} \geq 0 \tag{7.16}$$

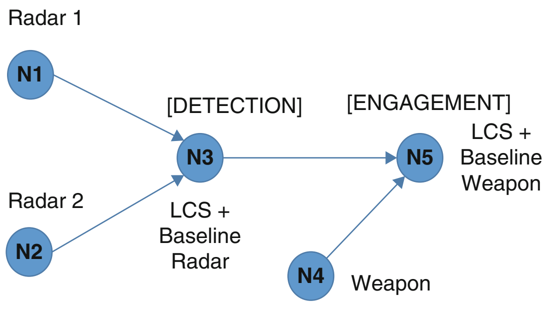
where p_{ij} , y_{ij} , z_{ij} are auxiliary variables and x is the vector of decision variables.

In general, linear formulations are amenable to highly efficient solvers that can handle very-large-scale problems with ease. In the context of our joint SODA-RPO framework, we leverage this ability to handle uncertainty in constraints by addressing the uncertainties of system requirements.

7.4 A Case Study: Naval Warfare Scenario

7.4.1 Example Problem Description and Data

We demonstrate our joint RPO-SODA framework through a simple case of a naval architecture scenario as exhibited in Fig. 7.4a, where the ability to engage a target is modeled as a sequential change of capabilities that are needed to accomplish the task of engagement. Figure 7.4b shows the SODA parameters that describe the connectivities between the nodes in terms of the SODA parameters of (SOD, IOD, and COD). Nodes N1 (Radar 1), N2 (Radar 2), and N4 (Weapon) are feeder nodes with yet-to-be determined systems from a choice catalogue of systems as described in Table 7.1.



$$SOD = \begin{bmatrix} 0 & 0 & 0.5 & 0 & 0 \\ 0 & 0 & 0.4 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.85 \\ 0 & 0 & 0 & 0 & 0.2 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$COD = \begin{bmatrix} 0 & 0 & 80 & 0 & 0 \\ 0 & 0 & 90 & 0 & 0 \\ 0 & 0 & 0 & 0 & 70 \\ 0 & 0 & 0 & 0 & 75 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$IOD = \begin{bmatrix} 0 & 0 & 100 & 0 & 0 \\ 0 & 0 & 50 & 0 & 0 \\ 0 & 0 & 0 & 0 & 80 \\ 0 & 0 & 0 & 0 & 66.67 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Fig. 7.4 (**a**) Naval scenario engagement architecture and (**b**) SODA matrices

Table 7.1 Architectural data and options

		SoS Capabilities (Outputs)		Capabilities (Outputs)		Requirements (Inputs)		Uncertainty		Cost
No.	System Name	R. Range	W. Range	[kW] Power.	Bandwidth [MBps]	[kW] Power	Bdwth [MBps]	Power	Bandwidth	[$]
1	Radar System 1	150	0	0	0	50	5	29.7	0.6	$100,000.00
2	Radar System 2	300	0	0	0	100	10	34.1	9.7	$20,000.00
3	Radar System 3	450	0	0	0	150	20	141.2	18.6	$300,000.00
4	Radar System 4	600	0	0	0	200	30	29.2	23.8	$400,000.00
5	Radar System 5	750	0	0	0	250	40	125.5	0.7	$500,000.00
6	Radar System 1	150	0	0	0	75	5	19.5	0.7	$500,000.00
7	Radar System 2	300	0	0	0	125	10	40.9	6	$650,000.00
8	Radar System 3	450	0	0	0	150	15	66.8	0.4	$750,000.00
9	Radar System 4	600	0	0	0	175	20	36.2	4.2	$850,000.00
10	Radar System 5	750	0	0	0	225	25	143.1	17.9	$900,000.00
11	Weapon System 1	0	100	0	0	0	0	0	0	$200,000.00
12	Weapon System 2	0	200	0	0	0	0	0	0	$300,000.00
13	Weapon System 3	0	300	0	0	0	0	0	0	$400,000.00
14	Weapon System 4	0	400	0	0	0	0	0	0	$450,000.00
15	Weapon System 5	0	500	0	0	0	0	0	0	$500,000.00
16	Control Station 1	0	0	100	10	0	0	92.4	9.4	$500,000.00
17	Control Station 2	0	0	150	20	0	0	85.9	0.8	$600,000.00
18	Control Station 3	0	0	175	30	0	0	36.4	27.5	$700,000.00
19	Control Station 4	0	0	200	40	0	0	150.4	3.1	$500,000.00
20	Control Station 5	0	0	475	50	0	0	203.5	27.7	$600,000.00

Table 7.1 shows the candidate list of systems that are available as feeder nodes [Radar systems 1–5 and Weapon systems 1–5], and support systems [Control Station 1–5] that are needed for the feeder node systems to be able to operate. Columns 1–2 are the capabilities of each available system and represent the operability of the system in performing its function on the network as shown in Fig. 7.4a. Columns 2–3 are the capabilities of support systems that are needed for the feeder nodes to operate. In this case, they reflect each of the types of control stations that can supply the communication bandwidth and power requirements to the choice of radar and weapons systems. Columns 5–6 are the requirements of power and bandwidth for the feeder nodes (Radar 1, Radar 2, Weapon) that need to be met through appropriate choice of control stations. Columns 7–8 are the symmetric uncertainties associated with the power and bandwidth requirements/capabilities for each system listed, for example, Radar System 1 has a power requirement of 50 kW (+/−) 29.7 kW and a bandwidth requirement of 5Mbps (+/ −) 0.6 Mbps. The final column lists the individual acquisition costs. In our simple example, the selection of the systems is constrained such that systems [1,8], [16,17], and [17,18] are mutually exclusive. Also, only 2 control systems can be selected at most. The motivation here is to determine the optimal collection of systems that should be acquired at N1, N2, and N4, given the piecewise linear interdependencies of the SODA network, such as to maximize the operability at the N5 node, under conditions of uncertainty in the power and bandwidth.

7.4.2 Results and Discussion

Figure 7.5a shows the resulting operabilities throughout the network, resulting in the final operability tradespace as a function of robustness choice in the power and communications related constraints of the resulting optimization problem. Note again that the current uncertainties are associated with system-level uncertainties at the feeder nodes and that the SODA network parameters remain constant. Table 7.2 shows a collection of portfolios at various points on the tradespace of Fig. 7.5b, corresponding to various levels of robustness (values of gamma) for the power and bandwidth.

The utility in the results lay not in the determination of a single optimal point, but rather to enable the practitioner a means of traversing the tradespace efficiently, and to determine candidate sets of systems that are deemed to be attractive. The portfolio results shown in Table 7.2, for example, may be a set of solutions that the practitioner deems to be of interest, and that the viewing of collections of systems will yield additional insights into selection. In this case, the major changes in architecture across the three portfolios chosen show that changes in selection of N2 mostly dictate the changes in performance across the portfolios. Given this, the practitioner may choose to acquire the other systems first and decide on choice of N2 later, with the knowledge at hand that certain types of uncertainties are considered in the decision.

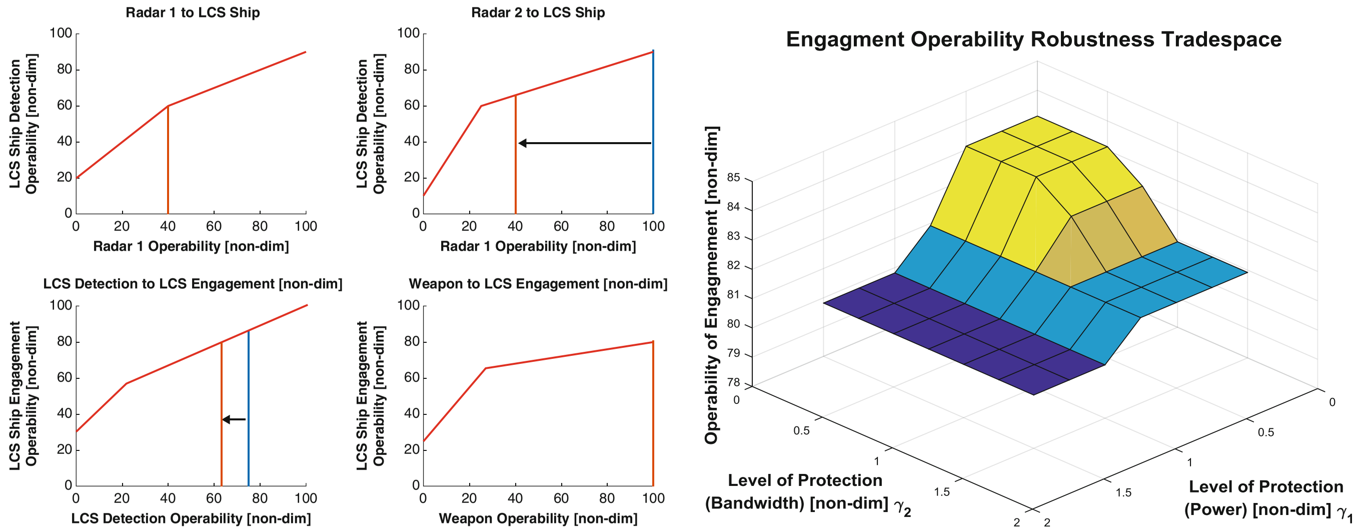


Fig. 7.5 Tradespace of operability using the joitn RPO–SODA analysis. Tradespace shown (*right*) illustrates the change in operability (O5). Impact also shown (*left*) on level of operability at each node and interdependency

Table 7.2 Portfolios at various points on tradespace

		No.	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	
Γ (Power)	Γ (Bwidth)	System Name	Radar System 1	Radar System 2	Radar System 3	Radar System 4	Radar System 5	Radar System 1	Radar System 2	Radar System 3	Radar System 4	Radar System 5	Weapon System 1	Weapon System 2	Weapon System 3	Weapon System 4	Weapon System 5	Control Station 1	Control Station 2	Control Station 3	Control Station 4	Control Station 5	Operability of Engagement (O5)
0	0	Portfolio 1	0	1	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	83.12
0.75	0.75	Portfolio 2	0	1	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	1	80.92
1.5	1.5	Portfolio 3	0	1	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	1	79.82

7.5 Conclusion

This chapter has presented an approach to managing system architectural development through a joint SODA-RPO approach. The approach introduces interdependency modeling capabilities of SODA, within the robust, optimization-based decision-making framework of RPO, to enable improved modeling and generation of architectural tradespace solutions. We demonstrate the joint framework with a simple naval scenario architectural problem of balancing local-level robustness characteristics in the architecture with the top-level performance in operability. An advantage of the approach is that it can include further operational characteristics of the SODA representation as piecewise linear functions within the mixed integer programming (MIP) problem formulation of RPO. Future work will entail a more general treatment of uncertainty in the joint RPO-SODA framework by exploring tractable means of including uncertainties in the SODA-generated interdependencies, for example.

Acknowledgments This material is based upon work supported, in whole or in part, by the U.S. Department of Defense through the Systems Engineering Research Center (SERC) under Contract HQ0034-13-D-005 RT #155. SERC is a federally funded University Affiliated Research Center managed by Stevens Institute of Technology.

References

1. Maier M (1998) Architecting principles for system-of-systems. Syst Eng 1(4):267–284
2. DeLaurentis D, Callaway R (2004) A SoS perspective for public policy decisions. Rev Policy Res 21(6):829–837
3. Yang K, Chen Y, Lu Y, Zhao Q (2010) The study of guided emergent behaviour in system of systems requirement analysis. International conference on System of Systems engineering
4. Hsu J (2009) Emergent behaviour of systems-of-systems. INCOSE mini-conference
5. DeLaurentis D (2005) Understanding transportation as a system-of-systems design problem. AIAA Aerospace Sciences Meeting and Exhibit
6. de Weck OL, Ross AM, Rhodes DH (2012) Investigating relationships and semantic sets amongst system lifecycle properties (ilities). Third International Engineering Systems Symposium
7. Davendralingam N, DeLaurentis D, Fang Z, Guariniello C, Han SY, Marais K, Mour A, Uday P (2014) An analytic workbench perspective to evolution of system of systems architectures. Conference on Systems Engineering Research (CSER)
8. Davendralingam N, DeLaurentis D (2015) A robust portfolio optimization approach to system of system architectures. Syst Eng 18(3):269–283
9. Guariniello C, DeLaurentis D (2016) Supporting design via the system operational dependency analysis methodology. Res Eng Des:1–17. doi:10.1007/s00163-016-0229-0
10. Garvey P, Pinto A (2012) Advanced risk analysis in engineering enterprise systems. CRC Press, Boca Raton
11. Garvey P, Pinto A, Santos JR (2014) Modelling and measuring the operability of interdependent systems and systems of systems: advances in methods and applications. Int J Syst Syst Eng 5(1):1–24
12. Guariniello C, DeLaurentis D (2014) Dependency analysis of system-of-systems operational and development networks. Conference on Systems Engineering Research (CSER)
13. Guariniello C, DeLaurentis D (2014) Integrated analysis of functional and developmental interdependencies to quantify and trade-off ilities for system-of-systems design, architecture, and evolution. Conference on Systems Engineering Research (CSER)
14. Bertsimas D, Sim M (2004) The price of robustness. Oper Res 52(1):35–53

Chapter 8
Interactive Model Trading for Resilient Systems Decisions

Adam M. Ross and Donna H. Rhodes

Abstract The Interactive Model-Centric Systems Engineering research effort is interested in developing knowledge necessary to leverage the increasing involvement of computational models in system design. A key activity in such model-centric environments is the selection and usage of models to generate data for decision making. Extending work from prior demonstrations, this paper presents a case study where insights are gained via usage and comparison of multiple models. The results highlight the need to explore how model choice and trade-off can impact the attractiveness of alternative systems. Sixteen tradespaces of cost-benefit data are generated via combined pairwise usage of four alternative evaluative models (performance and cost calculations) and four alternative value models (calculating the "goodness" of different levels of performance and cost). These tradespaces are used to determine attractive Pareto efficient Space Tug vehicles, as well as insights that are less sensitive to model choice. No best model is shown, but rather different models provide insights into different aspects of the system evaluation and valuation activity. Two categories of insights are highlighted: patterns in the structure of the decision problem (i.e., how "value" is defined and what systems might be feasible) and artifacts of the models themselves (i.e., how to mitigate against misleading results due to model abstractions).

Keywords Models • Modeling • Evaluative models • Value models • Fidelity • Robust solutions • Tradespace exploration • Resilience

8.1 Introduction

Data-driven decision making is an increasingly used phrase to describe a paradigm where evidence, often in the form of digital artifacts, is used to support a methodical approach to making decisions. Implied in this approach is a numerical basis, possibly supported by large databases of information, where the "data" is leveraged for compelling insights that would (presumably) result in better decisions (than a

A.M. Ross (✉) • D.H. Rhodes
Massachusetts Institute of Technology, 77 Massachusetts Ave, Cambridge, MA, USA
e-mail: adamross@mit.edu; rhodes@mit.edu

A.M. Madni et al. (eds.), *Disciplinary Convergence in Systems Engineering Research*, https://doi.org/10.1007/978-3-319-62217-0_8

less data-driven approach). In the world of systems engineering, especially early in the life cycle, where empirical data may be lacking, a model-based approach is increasingly being used. Such model-based approaches leverage computational techniques to generate *artificial* data about (potential) systems. The term *artificial* is not meant pejoratively, but rather to reflect the human-made abstraction that is used in order to generate the data [1]. Since models are abstractions, they are necessarily deficient to some degree in their predictive power to describe a system (form, function, behavior, operations, etc.) as it *actually will be experienced* later in the life cycle. Various efforts have been made in many fields to address concepts such as fidelity, accuracy, precision, and so on, all in support of selecting appropriate models for a decision at hand.

8.1.1 Motivation

The Interactive Model-Centric Systems Engineering research program has proposed that systems engineers should not just try to select models to answer questions, but also to better reflect upon what can be learned using different models to answer the same question. Two previous papers published at CSER have addressed the two main types of models used in early life cycle systems engineering: evaluative models and value models. The former tries to predict the performance and cost of potential alternative systems, while the latter tries to predict how much different levels of performance and cost *is worth* to different stakeholders. Both types of models require abstraction of reality, and both generate essential data needed for confident decision making. The prior papers have shown that using alternative model implementations simultaneously (i.e., not just picking one of them) in order to generate data can provide insights that can make the ultimate decisions more resilient to uncertainties and deficiencies inherent in the act of modeling itself. Such resiliency can be manifested in decisions that are insensitive to uncertainties, or can be changed at low cost to account for new information as it unfolds.

This paper synthesizes the prior studies into a combine model choice and trading study in order to determine what kinds of additional insights could be had when taking this broader perspective. In this study, the researcher has interacted both with model choice and through active exploration of the resulting datasets. Interactivity has been found to be a useful mechanism for hypothesis generation and testing with dataset displaying emergent properties and is a central topic of a current doctoral research effort [2].

8.2 Demonstration of Combined Value and Evaluative Model Trading: Space Tug

In order to demonstrate the effects of trading both value and evaluative models, and the insights that can be gained by doing so, we will return to the Space Tug case used in the prior demonstration of separately considered value model trading and evaluative model trading [3]. The generic mission is therefore the same but the key questions have now combined:

> A decision maker has a budget for an orbital transfer vehicle (a.k.a. "Space Tug") and thinks he knows what he wants (in terms of attributes of goodness of a system). But he is aware that he may not have formulated his value model correctly. Additionally, he is aware that Space Tugs are a developing technology, and the models used to evaluate them are not 100% validated. He wants to explore various uncertainties in his value model as well as a variety of evaluative model implementations in order to understand their impact on what makes a "good" system alternative and whether there are alternatives that are resilient to these uncertainties.

For the combined demonstration, we used both sets of value models (four implementations) and evaluative models (also four implementations) described in our prior demonstrations. These resulted in 16 tradespaces worth of data generated via the pairwise combination of value and evaluative models listed in Table 8.1. Value models included multi-attribute utility (MAU), analytic hierarchy process (AHP), cost-benefit analysis (CBA), and measure of effectiveness (MOE). Evaluative models included four implementations including the original (#1), new speed (#2), new material (#3), and combined new speed and material (#4). The details of these models can be found in their respective CSER papers [4, 5]. For each of the 16 tradespaces, we determined the Pareto set of alternatives (most efficient in benefit-cost trade-off) and interactively determined appropriate fuzziness levels such that alternatives became insensitive in attractiveness (i.e., Pareto set membership) due to model choice. Additional emergent insights that were captured are described below.

Table 8.1 Value and evaluative models used in demonstration

Evaluative model: value model	Implementation #1 (original)	Implementation #2 (new speed)	Implementation #3 (new material)	Implementation #4 (new speed and material)
MAU	1-MAU	2-MAU	3-MAU	4-MAU
AHP	1-AHP	2-AHP	3-AHP	4-AHP
CBA	1-CBA	2-CBA	3-CBA	4-CBA
MOE	1-MOE	2-MOE	3-MOE	4-MOE

8.3 Results

The results of this demonstration will be presented as a value model by evaluative model and evaluative model by value model comparison. That is, we repeated our value model trade-off study [4] for *each* of the evaluative models (4 value models × 1 evaluative model) × 4 evaluative models. Then we repeated our evaluative model trade-off study [5] for each of the value models (4 evaluative models × 1 value model) × 4 value models. We did this in order to demonstrate generalization of the approach we took within the prior studies, and to highlight the intent as one that aims to gain knowledge about the impact of model choice, rather than selection of the "best" alternative.

8.3.1 Value Model Trading for Each Evaluative Model

As described in the earlier value model trading case, there are four pairs of objectives considered when determining Pareto efficient design sets [4]. These are (1) MAU-COST, (2) AHP-COST, (3) CBA-COST, and (4) MOE-COST, with each value model resulting in a metric quantifying the expected benefit and cost of an alternative. Each value model has intentionally kept the cost metric as a distinct objective in order to explicitly highlight the various cost versus benefit trade-offs that determine value. This is reflected in the objective sets each having cost as well as the appropriate value model metric for benefit. Table 8.2 describes the size of the 0% fuzzy Pareto sets for each of the value models when using each of the evaluative model implementations (e.g., *Eval Model #1* is the original model). These sets represent the most efficient benefit-cost trade-off alternatives for a given value model-evaluative model pair tradespace. The joint set contains those alternatives that appear in all of the Pareto sets across the value models for a given evaluative model. The compromise set contains those alternatives in the Pareto set defined by considering all of the objectives simultaneously for a given evaluative model. For this study, each objective set was assigned to a notional decision maker (DM) to help conceptualize the study (i.e., what if four different DMs had four different value models, how do we find alternatives that will satisfy all (or most) of them?).

Table 8.2 Pareto sets' sizes for value models using each evaluative model implementation

	# designs in 0% Pareto set			
Objective set	Eval Model #1	Eval Model #2	Eval Model #3	Eval Model #4
1: MAU-cost	10	9	15	16
2: AHP-cost	43	10	54	19
3: CBA-cost	17	6	25	5
4: MOE-cost	13	13	20	20
Joint	0	0	0	0
Compromise	6	2	15	5

8.3.1.1 Evaluative Model #1 (Original Model)

Evaluative model #1 has a potential tradespace size of 384 designs, enumerated across three design variables (payload size, propulsion type, and fuel tank size). Table 8.3 lists the designs that are almost joint Pareto efficient (a.k.a. "promising designs), appearing in 3 out of 4 Pareto sets.

As one increases the degree of acceptable fuzziness (i.e., distance from the Pareto front), the number of designs in a given fuzzy Pareto set increases. The first fully joint (across 4 out of 4 value models) Pareto efficient design, which appears at a fuzzy level of 7%, is design 52, a medium payload, electric propulsion, and medium fuel tank design. Figure 8.1 illustrates the relative sizes of the fuzzy joint Pareto sets at 0% and 7% fuzzy levels.

Table 8.3 Promising designs that are joint Pareto efficient across 3 out of 4 value models in evaluative model #1

ID number	Pareto efficient for	Invalid for	Details
1	2, 3, 4	1	Small biprop min fuel
11	2, 3, 4	1	Small cryo near min fuel
63	1, 2, 3		Med nuke near max fuel
95	1, 2, 3		Large nuke near max fuel
127	1, 2, 3		Xl nuke near max fuel
128	1, 2, 3		Xl nuke max fuel

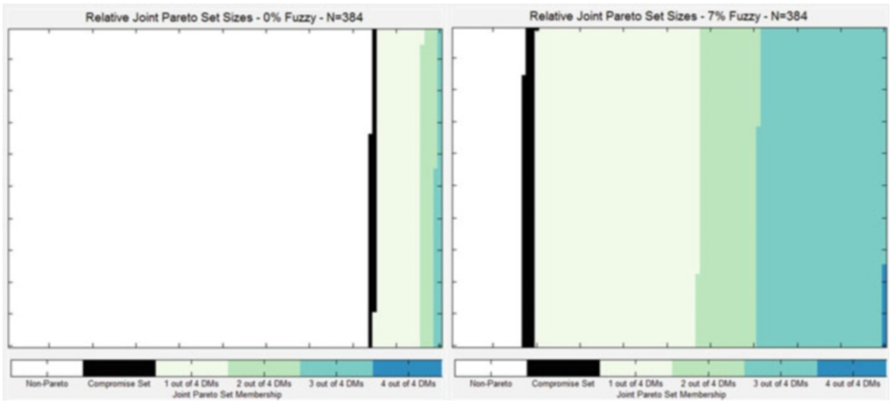


Fig. 8.1 Gridmap showing relative sizes of 0% (*left*) and 7% (*right*) fuzzy joint Pareto sets in evaluative model #1

8.3.1.2 Evaluative Model #2 (New Speed Model)

Evaluative model #2 has a potential tradespace size of 384 designs, enumerated across three design variables (payload size, propulsion type, and fuel tank size). This model implementation has a new (higher fidelity) speed model. Table 8.4 lists the designs that are almost joint Pareto efficient, appearing in 3 out of 4 Pareto sets.

The same design 52 appears as the first fully joint Pareto efficient at a 7% fuzzy level. For this evaluative model, it looks like the electric propulsion type passes the nuclear type in terms of "promising" for the new speed model. The first joint Pareto design appears at the same fuzzy level as the original evaluative model.

8.3.1.3 Evaluative Model #3 (New Material Model)

Evaluative model #3 has a potential tradespace size of 768 designs, enumerated across four design variables (payload size, propulsion type, fuel tank size, and material type). This model implementation added "fidelity" to the design space description, allowing for variation in the material type between aluminum and carbon. Table 8.5 lists the designs that are almost joint Pareto efficient, appearing in 3 out of 4 Pareto sets.

Table 8.4 Promising designs that are joint Pareto efficient across 3 out of 4 value models in evaluative model #2

ID number	Pareto efficient for	Invalid for	Details
1	2, 3, 4	1	Small biprop min fuel
9	2, 3, 4	1	Small cryo min fuel
87	1, 2, 3		Large elec near max fuel
119	1, 2, 3		Xl elec near max fuel
120	1, 2, 3		Xl elec max fuel

Table 8.5 Promising designs that are joint Pareto efficient across 3 out of 4 value models in evaluative model #3

ID number	Pareto efficient for	Invalid for	Details
1	2, 3, 4	1	Small biprop min fuel
11	2, 3, 4	1	Small cryo near min fuel
63	1, 2, 3		Med nuke near max fuel
95	1, 2, 3		Large nuke near max fuel
128	1, 2, 3		Xl nuke max fuel
512	1, 2, 3		Xl nuke max fuel CARBON

Two fully joint (across 4 out of 4 value models) Pareto efficient designs appear at a fuzzy level of 6%: design 52 (medium payload, electric propulsion, medium fuel tank, aluminum design) and design 435 (like 52, but with carbon and 1 size smaller fuel tank).

8.3.1.4 Evaluative Model #4 (New Speed and Material Model)

Evaluative model #4 has a potential tradespace size of 768 designs, enumerated across four design variables (payload size, propulsion type, fuel tank size, and material type). This model implementation incorporates both the (higher performance fidelity) new speed model as well as the (higher design fidelity) new material model. Table 8.6 lists the designs that are almost joint Pareto efficient, appearing in 3 out of 4 Pareto sets.

The same two fully joint Pareto efficient designs appear at fuzzy level 6% as in evaluative model #3: designs 52 and 435. Figure 8.2 illustrates the relative sizes of the fuzzy joint Pareto sets at 0% and 6% fuzzy levels.

Table 8.6 Promising designs that are joint Pareto efficient across 3 out of 4 value models in evaluative model #4

ID number	Pareto efficient for	Invalid for	Details
1	2, 3, 4	1	Small biprop min fuel
9	2, 3, 4	1	Small cryo min fuel
120	1, 2, 3		Xl elec max fuel
504	1, 2, 3		Xl elec max fuel CARBON

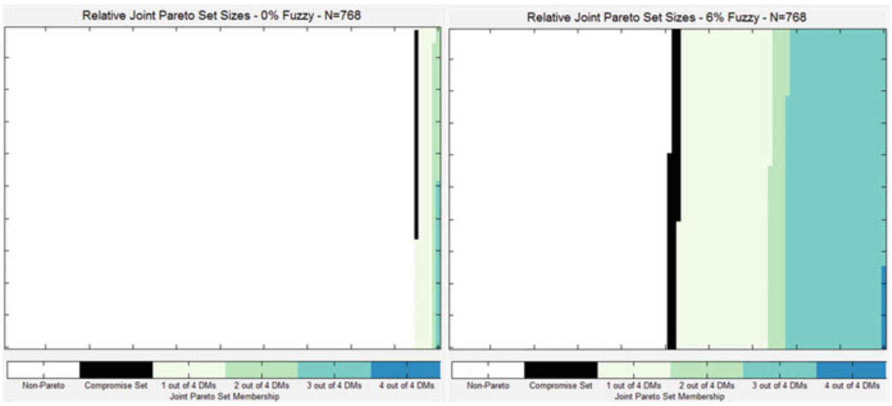


Fig. 8.2 Gridmap showing relative sizes of 0% (*left*) and 7% (*right*) fuzzy joint Pareto sets in evaluative model #4

The addition of having carbon as a material type affects the first joint Pareto design, as this matches model #3. But in that case the change from 7% to 6% is likely an artifact of the increased cost range due to the addition of very expensive carbon designs to the tradespace (i.e., fuzzy level is calculated as a fraction of the tradespace cost range, so the addition of more expensive designs will make a given design appear relatively closer to the Pareto front). The new speed model makes electric more promising than nuclear type propulsion, as both model #2 and model #4 drop all promising nuclear designs and add promising electric designs.

8.3.1.5 Summary Across the Evaluative Models

All four evaluative model implementations corroborate a tension between the MAU and MOE value models. That is, MOE value model prefers inexpensive, small designs in order to maximize delta-V, but none of those designs meet minimum acceptable performance levels for MAU, so they are considered invalid for the MAU value model. The MAU and MOE value models are in tension, as the MOE value model prefers low mass designs (strongly driven by small payloads). This can be seen in Fig. 8.3, which illustrates the single-attribute utility score as a function of payload capability across the four evaluative models. As payload capability goes up, so too does the mass of the alternatives. The red triangles indicate the designs that MOE views as Pareto efficient, while the blue triangles are the designs in the MAU Pareto set. Many red triangles are below the U0 (red) line in the figure, which correspond to the minimum acceptable level of capability for the utility model. The MAU model (which includes the illustrated single-attribute utility curves, as well as utility curves for other attributes) requires that payload capability be greater than the minimum possible size. This clearly shows that MOE and MAU value is in tension. This insight applies across all of the evaluative models and is an *insight about the nature of the value models*.

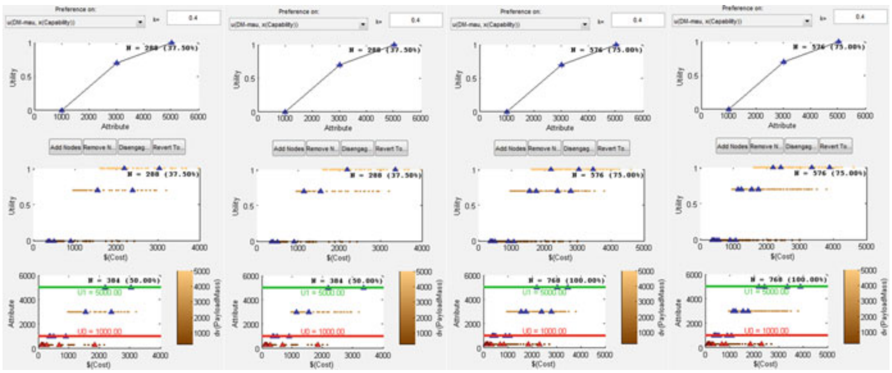


Fig. 8.3 Comparison of the payload capability single-attribute utility function impact across the four evaluative models with MAU (*blue triangle*) and MOE (*red triangle*) Pareto sets indicated

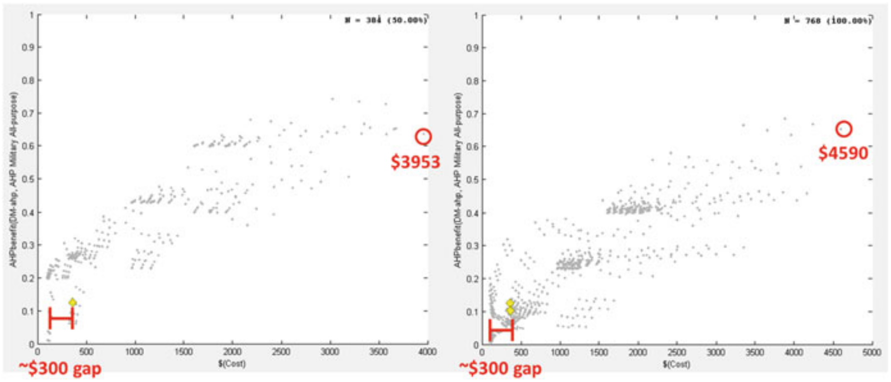


Fig. 8.4 Comparing evaluative model #1 first joint Pareto design at fuzzy level 7% (*left*) and evaluative model #4 first joint Pareto at fuzzy level 6% (*right*) illustrates how increase in maximum cost design impacts fuzzy metric

Another insight is that the minimum fuzzy level for the appearance of joint Pareto designs decreased from 7% to 6% when adding the new material model (in evaluative models #3 and #4). Design 52 is the key design in both cases (with design 435 also in models #3 and #4), and according to the analysis, these designs do not move much across the evaluative models. Instead, the change in fuzzy level required appears to be an artifact of both how the fuzzy Pareto number is calculated and having more expensive designs appearing in the tradespace. Figure 8.4 shows the increased cost range in the tradespace due to the addition of higher cost designs (the new more expensive carbon version designs in models #3 and #4). This is an example of a *modeling artifact* and not an insight about the designs themselves.

8.3.2 *Evaluative Model Trading for Each Value Model*

For this part of the case, we look at the tradespaces for a given value model across the four different evaluative models [5] (essentially repeating the early evaluative model trade case for each of the four value models). For the Pareto set category tables, the categories are different than in the corresponding Pareto sets described in the earlier evaluative model trading case paper. The categories are consistent across the examples below, however. These categories range from A to K, with each corresponding to different patterns of Pareto set membership across models.

8.3.2.1 Multi-attribute Utility Model

Looking across the evaluative models within the MAU value model is what was done in the 2016 CSER paper (evaluative model trading). Table 8.7 below

Table 8.7 Designs, marked in gray with check, for model implementations in which they are efficient for the MAU value model

Category	Design ID (Aluminum)	*Eval Model #1*	*Eval Model #2*	*Eval Model #3*	*Eval Model #4*	Design ID (Carbon)	*Eval Model #3*	*Eval Model #4*
A	52, 53, 63	✓	✓	✓	✓	436, 437, 447	✓	✓
B	54, 87, 119	✓	✓	✓	✓	438, 471, 503		✓
D	86, 120		✓		✓	470, 504		✓
E	96, 128	✓		✓		480, 512	✓	
F	127	✓				511		
K	95	✓	✓	✓		479	✓	

A. Designs 52, 53, and 63 (and their carbon counterparts) are always efficient across the model implementations

B. Designs 54, 87, and 119 are efficient across the model implementations, and their carbon counterparts are only efficient under the new speed model (model #4)
D. Designs 86 and 120 are efficient only under the new speed model (models #2 and #4), as are their carbon counterparts (in model #4)
E. Designs 96 and 128 are efficient only under the old speed model (models #1 and #3), as are their carbon counterparts (in model #3)
F. Design 127 is efficient only under the original model (model #1)
K. Design 95 (and its carbon counterpart) is efficient under each of the models except the combined model (model #4)

summarizes the results of which designs are Pareto efficient for a given evaluative model. Six categories of designs are apparent in this table.

8.3.2.2 Analytic Hierarchy Process Model

The next value model, AHP, was then compared across the evaluative models. First we identified designs that are in the Pareto set within each model. These designs are described in Table 8.8. Ten categories of designs are apparent in this table.

As can be seen in Fig. 8.5, electric vehicles (blue) replace nuclear vehicles (green) on the Pareto front under the new speed model (old speed models #1 and #3 on the left, and new speed models #2 and #4 on the right).

8.3.2.3 Cost-Benefit Analysis Model

The next value model, CBA, was then compared across the evaluative models. First we identified designs that are in the Pareto set within each model. These designs are described in Table 8.9. Six categories of designs are apparent in this table.

Similar to the pattern for the AHP value model, Fig. 8.6 shows that electric vehicles replace nuclear vehicles on the Pareto front under the new speed model.

Table 8.8 Designs, marked in gray with check, for model implementations in which they are efficient for the AHP value model

Category	Design ID (Aluminum)	*Eval Model #1*	*Eval Model #2*	*Eval Model #3*	*Eval Model #4*	Design ID (Carbon)	*Eval Model #3*	*Eval Model #4*
A	22	✓	✓	✓	✓	406	✓	✓
B	9	✓	✓	✓	✓	393		✓
C	127	✓	✓	✓	✓	511	✓	
D	21, 23, 87, 118, 119, 120		✓		✓	405, 407, 471, 502, 503, 504		✓
E	10, 11, 12, 13, 14, 31, 77, 94, 109, 126, 128	✓		✓		394, 395, 396, 397, 398, 415, 461, 478, 493, 510, 512	✓	
F	4, 5	✓				388, 389		
G	55					439		✓
H	1	✓	✓	✓	✓	385		
I	30	✓				414	✓	
J	2, 3, 62, 63, 65, 66, 67, 68, 69, 73, 74, 75, 76, 93, 95, 97, 98, 99, 100, 101, 105, 106, 107, 108, 125	✓		✓		386, 387, 446, 447, 449, 450, 451, 452, 453, 457, 458, 459, 460, 477, 479, 481, 482, 483, 484, 485, 489, 490, 491, 492, 509		

A. Design 22 (and its carbon counterpart) is always efficient across the model implementations

B. Design 9 is efficient across the model implementations, and its carbon counterpart (design 393) is only efficient under the new speed model (model #4)
C. Design 127 is efficient across the model implementations, and its carbon counterpart (design 511) is only efficient under the old speed model (model #3)
D. Designs 21, 23, 87, 118, 119, and 120 are efficient only under the new speed model (models #2 and #4), as are their carbon counterparts (in model #4)
E. Designs 10, 11, 12, 13, 14, 31, 77, 94, 109, 126, and 128 are efficient only under the old speed model (models #1 and #3), as are their carbon counterparts (in model #3)
F. Designs 4 and 5 are efficient only under the original model (model #1)
G. Design 439 is the carbon version of design 55, but is only efficient in the carbon variant under the new speed model (model #4)
H. Design 1 is efficient across all of the model implementations, but its carbon variant is never efficient
I. Design 30 is only efficient in the original model (model #1) and its carbon counterpart replaces it as efficient under the old speed model (model #3)
J. These designs are only efficient in the old speed model (model #1 and model #3), but their carbon variants are never efficient

8.3.2.4 Measure of Effectiveness Model

The next value model, MOE, was then compared across the evaluative models. First, we identified designs that are in the Pareto set within each model. These designs are described in Table 8.10. Two categories of designs are apparent in this table.

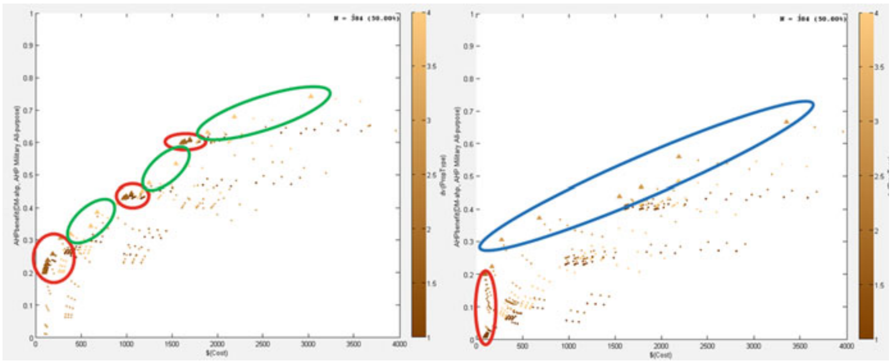


Fig. 8.5 AHP tradespace scatterplot under the old speed models (#1 and #3, *left*) and new speed models (#2 and #4, *right*), with families of designs indicated by propulsion type (*red* = biprop/cryo, *green* = nuclear, *blue* = electric)

Table 8.9 Designs, marked in gray with check, for model implementations in which they are efficient for the CBA value model

Categor y	Design ID (Aluminum)	*Eval Model #1*	*Eval Model #2*	*Eval Model #3*	*Eval Model #4*	Design ID (Carbon)	*Eval Model #3*	*Eval Model #4*
D	9, 120		✓		✓	393, 504		✓
E	11, 12, 13, 14, 29, 30, 31, 63, 95, 96, 128	✓		✓		395, 396, 397, 398, 413, 414, 415, 447, 479, 480, 512	✓	
F	4, 5, 127	✓				388, 389, 511		
H	1	✓	✓	✓	✓	385		
J	3, 32	✓		✓		387, 416		
L	87, 88, 119		✓			471, 472, 503		

D. Designs 9 and 120 are efficient only under the new speed model (models #2 and #4), as are their carbon counterparts (in model #4)

E. Designs 11, 12, 13, 14, 29, 30, 31, 63, 95, 96, and 128 are efficient only under the old speed model (models #1 and #3), as are their carbon counterparts (in model #3)
F. Designs 4, 5, and 127 are efficient only under the original model (model #1)
H. Design 1 is efficient across all of the model implementations, but its carbon variant is never efficient
J. These designs are only efficient in the old speed model (model #1 and model #3), but their carbon variants are never efficient
L. Designs 87, 88, and 119 are only efficient in the new speed model (model #2) without carbon as an option (model #3)

Figure 8.7 shows the Pareto set in evaluative models #1 and #2 (left) in green triangles, and in models #3 and #4 (right) with the red triangles corresponding to the new carbon version designs added in models #3 and #4. Notice how the sets do not change much, as the red points are just minor cost-increased version of their

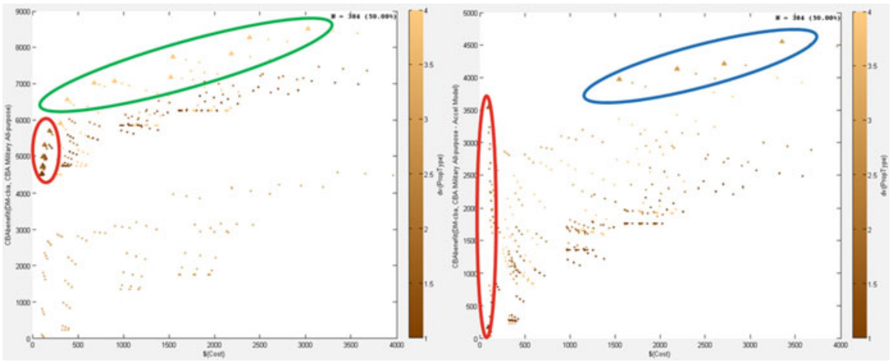


Fig. 8.6 CBA tradespace scatterplot under the old speed models (#1 and #3, *left*) and new speed models (#2 and #4, *right*), with families of designs indicated by propulsion type (*red* = biprop/cryo, *green* = nuclear, *blue* = electric)

Table 8.10 Designs, marked in gray with check, for model implementations in which they are efficient for the MOE value model

Category	Design ID (Aluminum)	*Eval Model #1*	*Eval Model #2*	*Eval Model #3*	*Eval Model #4*	Design ID (Carbon)	*Eval Model #3*	*Eval Model #4*
A	18, 19, 20, 21, 22, 23, 24	✓	✓	✓	✓	402, 403, 404, 405, 406, 407, 408	✓	✓
H	1, 2, 9, 10, 11, 17	✓	✓	✓	✓	385, 386, 393, 394, 395, 401		

A. Designs 18, 19, 20, 21, 22, 23, and 24 (and their carbon counterparts) are always efficient across the model implementations

H. Designs 1, 2, 9, 10, 11, and 17 are efficient across all of the model implementations, but their carbon variants are never efficient

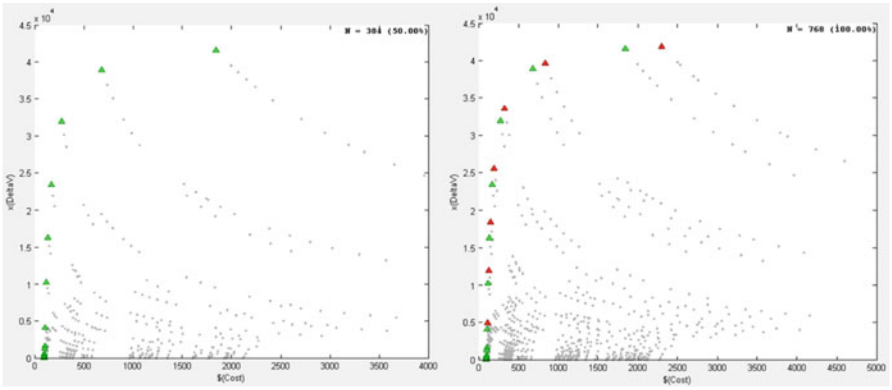


Fig. 8.7 Comparison of MOE value model across evaluative models #1 and #2 (*left*) and models #3 and #4 (*right*)

aluminum counterparts in green. This is due to the fact that this MOE (delta-V) is unaffected by the change in the model implementations relative to the original model. Carbon becomes Pareto efficient when cost gets high enough (i.e., not for the first couple of small designs). Small payloads dominate delta-V.

8.4 Discussion of Combined Value and Evaluative Model Trading

The key takeaway from the combined value and evaluative model case is that doing this kind of activity can reveal two classes of insights:

1. Insights into the fundamental relationship between perspectives of value and what is possible (*structural patterns for the decision problem*)
2. Insights into modeling artifacts, both in how value is captured and how evaluation is performed (*modeling artifacts*)

The first class of insights sometimes appears to emerge through the course of analysis. This may be due to the fact that the relationships are buried in the interactions between factors of the problem and are not readily apparent in our mental models. For example, Fig. 8.8 shows the CBA value model scatterplot in the evaluative model #4 (combined new material and new speed model). Lines show Pareto efficient points connected to similar designs with different levels of fuel. For both the electric and biprop propulsion types, there is a positive trade-off of more fuel (= more cost) for more benefit. But counterintuitively the small cryo propulsion designs actually get less benefit for more fuel. This is because the added fuel actually decreases the speed of those small designs in spite of increasing the onboard delta-V. This is a consequence of the confluence of physics (i.e., the rocket equation and inertia) and expectations on what is considered beneficial. The very fact that the relationship between fuel mass and benefit plays out differently in different regions of the tradespace means that the complexity (in terms of the number of factors to consider) likely would overwhelm an unaided human mind due to bounded rationality.

Each evaluative model is one representation of how a system might "perform," while each value model is one representation of how a system might "be valued." The emergence described above would occur at the intersection of each possible evaluative and value model, as well as across them, as shown in this simple Space Tug demonstration. Systems engineers and analysts may benefit strongly by considering not only their choice of evaluative and value models but also how their insights might vary if they were to include more than one of each type of model.

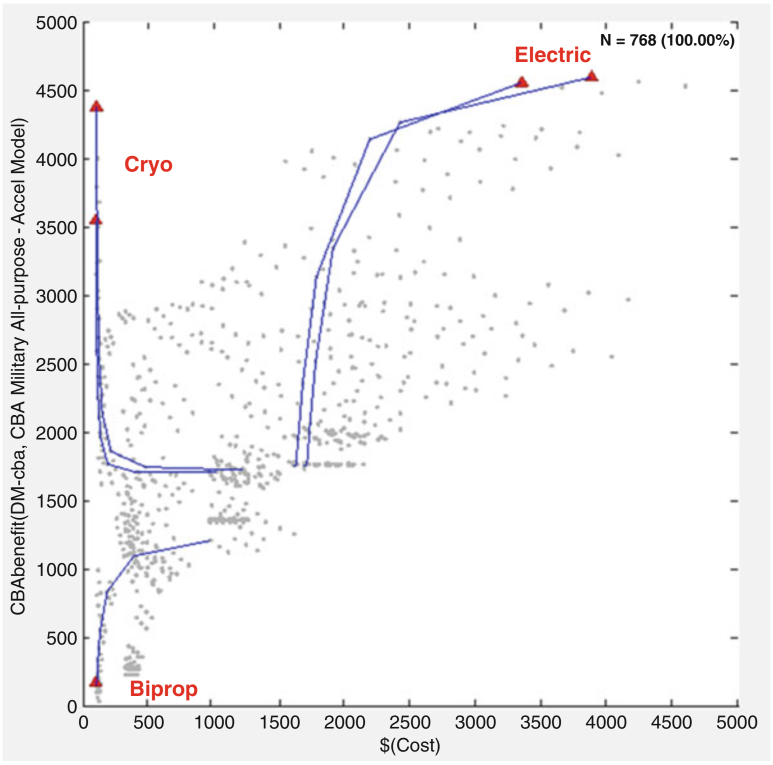


Fig. 8.8 CBA value model scatterplot for evaluative model #4 (combined new material and new speed)

Acknowledgments The authors would like to thank Dr. Matthew Fitzgerald for prior work. This material is based upon work supported, in whole or in part, by the US Department of Defense through the Systems Engineering Research Center (SERC) under Contract HQ0034-13-D-0004. SERC is a federally funded University Affiliated Research Center managed by the Stevens Institute of Technology. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the US Department of Defense.

References

1. Simon HA (1996) The sciences of the artificial. MIT Press, Cambridge, MA
2. Curry M, Ross AM (2016) Designing system value sustainment using interactive epoch-era analysis: a case study for on-orbit servicing vehicles. 14th Conference on Systems Engineering Research, Huntsville, AL, March 2016

3. McManus H, Schuman T (2003) Understanding the orbital transfer vehicle trade space. AIAA Space 2003, Long Beach, CA; September 2003
4. Ross AM, Rhodes DH, Fitzgerald ME (2015) Interactive value model trading for resilient systems decisions. 13th Conference on Systems Engineering Research, Hoboken, March 2015
5. Ross AM, Fitzgerald ME, Rhodes DH (2016) Interactive evaluative model trading for resilient systems decisions. 14th Conference on Systems Engineering Research, Huntsville, March 2016

Chapter 9
An Empirical Study of Technical Debt in Open-Source Software Systems

Reem Alfayez, Celia Chen, Pooyan Behnamghader, Kamonphop Srisopha, and Barry Boehm

Abstract Technical debt (TD) is a term coined by agile software pioneer Ward Cunningham to account for the added software-system effort or cost resulting from taking early software project shortcuts. The debt metaphor reflects that debt accumulates interest: the later it is paid, the more it costs. The TD concept has achieved extensive visibility and usage in the software field, but it applies at least as strongly to cyber-physical systems. In researching the TD phenomena, we have found that open-source software projects are particularly good subjects, as they keep records of the timing, content, and rationale for each update. In this paper, we concentrate on the analysis of open-source software projects to evaluate the relationships between multiple software system characteristics and TD and the relationships between software process factors and TD.

Keywords Open-source software • Software domain • Software metrics • Software quality • Software size • Technical debt

9.1 Introduction

Technical debt (TD) was defined by Cunningham [1] as the added software-system effort or cost resulting from taking early software project shortcuts or when short-term needs are addressed first. As with financial debt, it accumulates interest: the later it is paid, the more it costs. Clearly, the debt metaphor applies at least as strongly to cyber-physical systems, with even greater impact if the recovery involves millions of automobiles or exploding cellphones.

There are more and less responsible ways of incurring technical debt. The more responsible way to treat technical debt as an investment is to enable more rapid delivery in time-critical situations, such as competing to become first-to-market for

R. Alfayez (✉) • C. Chen • P. Behnamghader • K. Srisopha • B. Boehm
Center for Systems and Software Engineering, University of Southern California, Los Angeles, CA, USA
e-mail: alfayez@usc.edu; qianqiac@usc.edu; pbehnamg@usc.edu; srisopha@usc.edu; boehm@usc.edu

A.M. Madni et al. (eds.), *Disciplinary Convergence in Systems Engineering Research*, https://doi.org/10.1007/978-3-319-62217-0_9

a new product, rapid fielding of defenses to terrorist threats, or trying to build the riskiest parts first to determine whether continuing the project is feasible. The less-responsible ways often suffer from a lack of foresight, such as in building the easiest parts first, neglecting nonfunctional requirements, or skimping on systems engineering or maintainability preparation.

A particularly attractive area for researching the causes and effects of incurring TD is in the open-source software area. This form of software development involves distributed individuals or groups of developers participating in concurrently developing and improving software systems, many of which (e.g., Linux, Apache, JBoss, and Tomcat) are reliably used around the world. They are a rich source of consistent data, as they keep records of the timing, change content, and rationale for each upgrade.

In this paper, we perform an empirical analysis study on Apache Java systems to understand how the system characteristics such as domain and size relate to its TD and TD density, and how system process factors relate to its TD and TD density. Our goal is to get an understanding of the relations between these factors to provide guidance for decision makers and system engineers to incorporate it into their trade-space studies. Section 2 summarizes the research approach and key analysis tools. Section 3 elaborates on the research questions and data collection choices. Section 4 summarizes the data analysis and results. Section 5 presents the threats to validity. Section 6 the related work. Section 7 the conclusions and planned future work. Lastly, the acknowledgments.

9.2 Background

9.2.1 Technical Debt

Definition: Technical debt was originally defined as "not quite right code which we postpone making it right." [2] As the metaphor evolved, it has been used to describe other kinds of debts such as test debt, personal debt, design debt, requirement debt, documentation debt, etc. [2, 3].

Measurement and Calculation: Technical debt is measured using the following software qualities [4]:

- Robustness describes how stable the application is and its ability to recover from failure.
- Performance efficiency measures how the application uses its resources efficiently and how responsive it is.
- Security indicates the system's ability to protect confidential information and prevent unauthorized intrusions.
- Transferability measures the ease with which a new team can understand the software and become productive.
- Changeability measures the software adaptability and modifiability.

Technical debt is calculated using two main components: the principal and the interest [5]. The principal measures the cost or effort for refactoring a quality rule violation, for example, the effort to change a meaningless variable name to a meaningful name. The interest measures the decreased productivity or extra defects occurrence due to violating a software quality rule or not fixing a violated rule.

In this study, we measured technical debt principal using SonarQube (http://www.sonarqube.org/). SonarQube implements the SQALE method, which is the leading-edge method developed by DNV ITGS France to assess technical debt while conforming to the ISO 9126 standard for software quality [6]. SonarQube uses the following formula to measure technical debt:

$$\begin{aligned}\text{Debt(in man days)} = {} & \text{cost\_to\_fix\_duplications} \\ & + \text{cost\_to\_fix\_violations} \\ & + \text{cost\_to\_comment\_public\_API} \\ & + \text{cost\_to\_fix\_uncoverd\_complexity} \\ & + \text{cost\_to\_bring\_complexiy\_below\_threshold}\end{aligned}$$

where

$$\begin{aligned} Duplications &= cost\_to\_fix\_one\_block^{*}\, duplicated\_blocks \\ Violations &= cost\_to\ fix\_one\_violation^{*}\, mandatory\_violations \\ Comments &= cost\_to\_comment\_one\_API^{*}\, public\_undocumented\_api \\ Coverage &= cost\_to\_cover\_one\_of\_complexity \\ & \quad {}^{*}uncovered\_complexity\_by\_tests (80\% of\ coverage\ is\ the\ objective) \end{aligned}$$

$$\begin{aligned} Complexity = {} & cost\_to\_split\_a\_method \\ & * (function\_complexity\_distribution >= 8) \\ & + cost\_to\_split\_a\_class \\ & * (class\_complexity\_distribution >= 60) \end{aligned}$$

After retrieving technical debt, we calculated technical debt density using the following formula:

$$\frac{TechnicalDebtDensity = TechnicalDebt\,(in\_man\_hours)}{KLOC}$$

9.3 Empirical Study Setup

In order to provide decision makers with guidelines to assist them in the decision making process, we want to understand how different system characteristics and system process factors relate to technical debt. We studied the relations between each factor and the total system technical debt to assess their relations on a system level, and we calculated technical debt density by normalizing the numerator total technical debt with the denominator KLOC in order to evaluate the correlations of the different variables in our study and the quality per capita in our subject systems.

9.3.1 Research Questions

- *RQ1*: Does source code size relate to the total technical debt and the technical debt density?
- *RQ2*: Do the total technical debt and the technical debt density in software vary among domains?
- *RQ3*: Do system process factors, including the number of commits, releases, branches, and contributors, relate to the total technical debt and the technical debt density?

9.3.2 Data Collection

Data was collected from the Apache Software Foundation (http://www.apache.org), and only Java systems were considered for this empirical study since there are sufficient numbers of Java systems from different domains. The following six software domains of Apache systems were selected in this paper:

- Big Data
- Database
- Library
- Network Server
- Web Framework
- XML

Tables 9.1 and 9.2 show the characteristics of the selected systems in each domain.

The data collection process involved establishing and applying consistent criteria for inclusion of well-known systems to ensure the quality of this study. We excluded systems that fall under multiple domains or systems that have empty repositories. Source code under example, sample, and tutorial folders were also excluded. Systems that fall under all of the following criteria were considered:

- There are more than one official releases.
- The latest stable release source code is available.
- The source code has well-established sizing.
- The source code is fully accessible.

Table 9.1 Characteristics of system data source

Domains	Number of systems	Average LOC
Big Data	16	44,992
Database	13	52,610
Library	35	113,612
Network Server	9	20,624
Web Framework	11	31,164
XML	7	51,569

Table 9.2 Classification of number of systems by LOC in each domain

Domains	[1999]	[1000,5000]	[5001,10,000]	>10,000
Big Data	0	1	0	15
Database	0	0	2	11
Library	2	11	7	15
Network Server	0	2	3	4
Web Framework	0	0	2	7
XML	0	0	3	4

9.3.3 Data Collection Challenges

Different open-source systems have different folder structures. While some systems reconcile with the standard conventions, other systems do not have any clear folder naming schema or folder structure. We overcame this challenge by reading system documentations and manuals, looking into commit logs, and reading the release notes. By taking these steps, we were able to exclude files that are not source code such as examples and tutorials. These steps are necessary to insure that we only measured source code technical debt.

9.4 Data Analysis and Results

9.4.1 Evaluation of Size Hypothesis

In order to reject the null hypothesis and support the size hypothesis, we clustered the data using various clustering algorithms to examine whether total technical debt and technical debt density differ across different size clusters.

Cluster analysis: we used K-means clustering analysis to group the data into clusters based on the size of systems. K-means clustering does not provide any predicted outcomes, rather the algorithm is intended to find patterns in the data and cluster them based on their similarity. Here we clustered the data based on the size of systems to examine whether total technical debt and the technical debt density differ significantly among each cluster.

Since K-means clustering requires a specific number of clusters to be generated, we need to know the optimal number of clusters in the dataset. The proposed solution is to first compute a clustering algorithm of interest using different values of clusters k. Next, the Silhouette width is drawn according to the number of clusters. The location of a peak is generally considered as an indicator of the appropriate number of clusters. Two clusters were suggested for both datasets.

Total technical debt clustering results: the results of K-means clustering consist of 2 clusters of sizes 11 and 80, respectively. Table 9.3 lists the means of technical debt and means of LOC of each cluster, and Fig. 9.1 visually represents the cluster solution.

Technical debt density clustering results: the results of K-means clustering consist of 2 clusters of sizes 11 and 80, respectively. Table 9.4 lists the means of technical debt density and means of LOC of each cluster, and Fig. 9.2 visually represents the cluster solution.

Overall, larger systems had more technical debt in total but less technical debt density, while smaller systems had less technical debt in total but higher technical debt density. This may seem like a surprise, as one would expect that larger systems would be harder to understand, leading to higher debt density.

Table 9.3 Cluster means

Cluster	Technical debt	LOC mean
	Mean	
1	2074.1818	120362.64
2	346.2475	18070.42

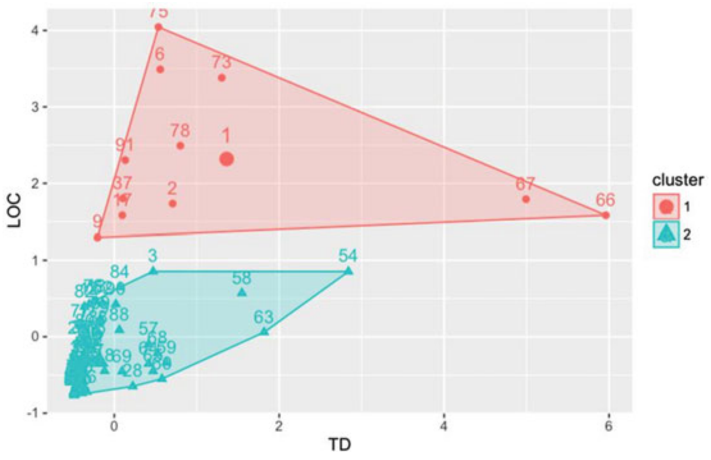


Fig. 9.1 2D representation of the cluster solution of technical debt

Table 9.4 Cluster means

Cluster	Technical debt	LOC mean
	Density mean	
1	19.48511	120,362.64
2	20.06992	18,070.42

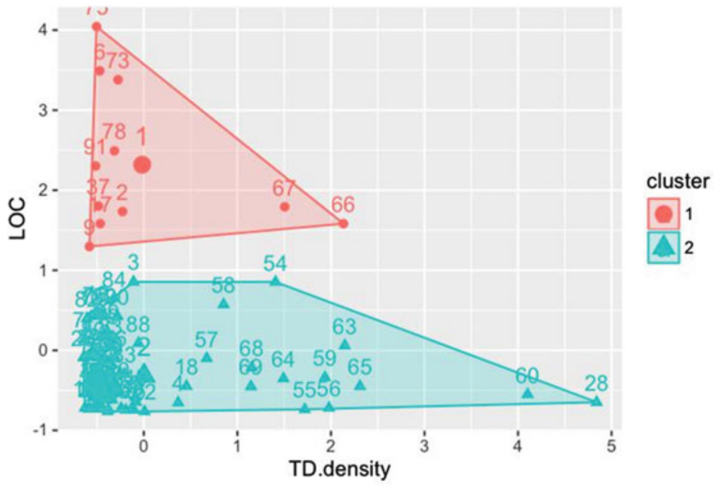


Fig. 9.2 2D representation of the cluster solution of technical debt density

However, there may be other explanations, at least for this sample. It could be that the larger projects had more capable teams and a more serious concern with high quality [7], or it could be that the larger projects had a larger proportion of simple, easy to understand components.

9.4.2 *Evaluation of Domain Hypotheses*

In this section, we performed various statistical analyses to examine whether total technical debt and technical debt density differ among different software domains.

9.4.2.1 Levene's Test

In order to perform one-way ANOVA to examine whether technical debt and technical debt density of the six domains differ significantly, we first performed Levene's test to examine whether the variances of the six domains are significantly different. Levene's test result indicates unequal variances ($F = 6.117$, $p = 6.912\text{e} - 05$) for total technical debt and unequal variances ($F = 4.9892$,

$p = 4.695e - 04$) for technical debt density. Since the p-value of Levene's test is much less than 0.05, we concluded that the variances of the six domains are significantly different, thus variances are unequal across domains. This would mean that we violated one of the assumptions of one-way ANOVA. Therefore, we could not perform one-way ANOVA to see if technical debt and technical debt density differ across domains. However, there are a number of ways to deal with unequal variances, such as transformations, robust regression, and so on. In this study, we used Welch ANOVA [8].

9.4.2.2 Welch ANOVA

The Welch ANOVA is based on the usual ANOVA F-test. However, the means are weighted by the reciprocal of the group mean variances. Therefore, it is suitable to handle unequal variances. An alpha level of 0.05 was used for all subsequent analyses.

The Welch ANOVA of total technical debt of the six different domains reveals a statistically significant difference, Welch's $F(5, 23.508) = 4.2964$, $p = 0.006408$, indicating that not all domains have the same total technical debt. The estimated omega squared ($\omega^2 = 0.15$) indicates that approximately 15% of the total variation in total technical debt is attributable to differences among the six domains.

The Welch ANOVA of technical debt density of the six different domains also reveals a statistically significant difference, Welch's $F(5, 25.47) = 5.2781$, $p = 0.001848$, indicating that not all domains have the same technical debt density. The estimated omega squared ($\omega^2 = 0.19$) indicates that approximately 19% of the total variation in technical debt density is attributable to differences among the six domains.

In conclusion, we have enough evidence to reject the null hypothesis and accept that both technical debt and technical debt density vary across the different domains.

9.4.2.3 Games-Howell Test

We concluded that at least two of the six domains differ significantly through Welch's ANOVA analysis; however, beyond that, we still couldn't tell the differences between all unique pairwise comparisons. Therefore, we performed a Games-Howell test to figure out which parings differ significantly.

Post hoc Games-Howell results indicate that systems in Library and Big Data have significantly different total technical debt at the 0.1 level of significance, while other comparisons are not significant. However, more pairings appear to have significantly different technical debt density at the 0.1 level of significance, including XML with Network, Web Framework with Network, Web Framework with Database, Library with Network, Network with Big Data, and Database with Big Data.

The variation of technical debt and technical debt density with different domains might be due to the variation in domains' complexity levels, development methods, personnel, and specialists [9]. A further work will investigate what factors contribute to technical debt variation more.

9.4.3 Evaluation of System Process Factors Hypotheses

In this section, we performed Pearson correlation tests to examine how different process factors relate to technical debt and technical debt density.

In order to understand how system process factors relate to technical debt and technical debt density, we used Pearson correlation to assess the relationships between each factor and the total technical debt and technical debt density. The significance level was set to 0.05, which is equal to a confidence level of 95%. Any p-value that is well below that threshold is concluded as a strong relationship.

Figure 9.3 shows the correlation between total technical debt and the system process factors. Table 9.5 lists the correlation coefficients and the significance levels of each system process factors and technical debt. While the results show a strong positive correlation between the number of commits and the total technical debt and the number of releases and the total technical debt, there is no significant correlation between the number of contributors and technical debt and the number of branches and technical debt.

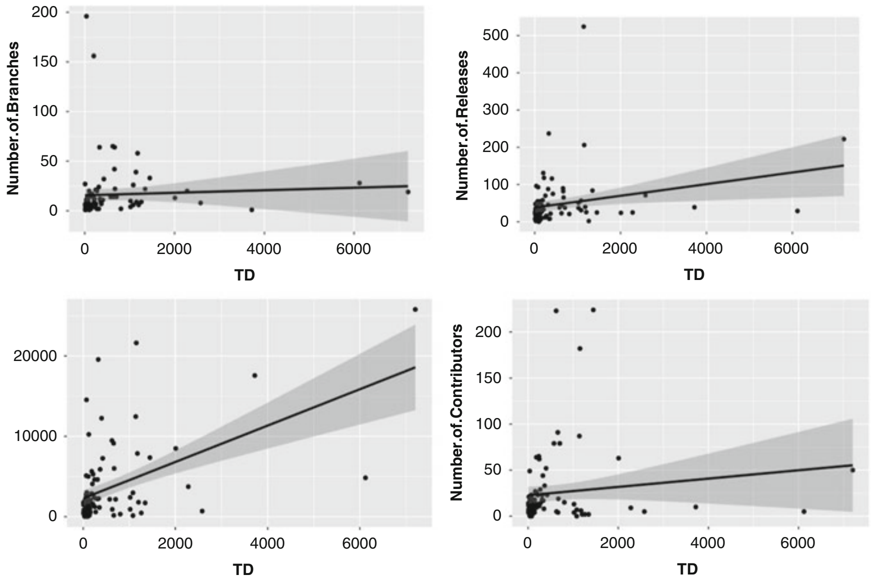


Fig. 9.3 TD and system process factors correlation

Table 9.5 Correlation coefficient matrix between total technical debt and system process factor

	N	R-value	p-value
Number of branches	91	0.05071	0.63308
Number of releases	91	0.26114	0.01241
Number of commits	91	0.51619	1.63135e-7
Number of contributors	91	0.12618	0.23335

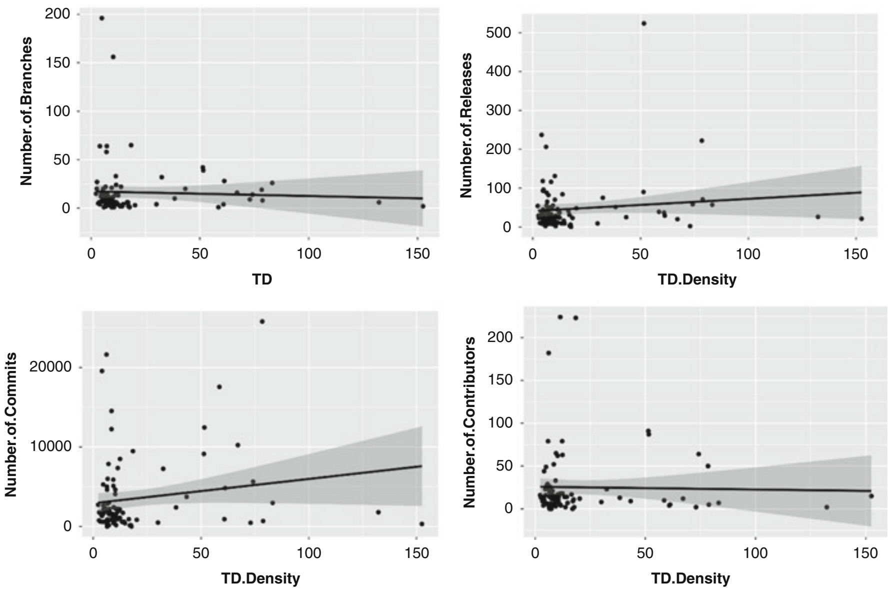


Fig. 9.4 TD density and system process factors correlation

Table 9.6 Correlation coefficient matrix between technical debt density and system process factors

	N	R-value	p-value
Number of branches	91	−0.04658	0.66108
Number of releases	91	0.12826	0.22567
Number of commits	91	0.17108	0.10493
Number of contributors	91	−0.02188	0.83688

Figure 9.4 shows the correlation between technical debt density and the system process factors. Table 9.6 shows that there is no significant correlation between technical debt density and any of the system process factors. This could be a side effect of the counter-intuitive results above on the correlations between larger and smaller systems and technical debt.

9.5 Threats to Validity

The key threats to external validity involve our subject systems. Although we only used 91 Java systems from six different domains, we selected the most popular systems and domains in one of the leading open-source software communities. The systems also vary along multiple dimensions, such as the number of versions, size, domain, and timeframe. The other domains do not have enough Apache Java projects that satisfy the selection criteria of our systems.

Further, we were unable to find sources of data outside of the open-source community to test hypotheses about the effects of various forms of closed-source process strategies on technical debt. Some companies are performing such studies, but generally prefer to keep the results private.

The construct validity of our study is mainly threatened by the accuracy of the amount of technical debt. To mitigate this threat, we chose SonarQube which is one of the most trusted tools to calculate technical debt. SonarQube is widely used and, to the best of our knowledge, is the only open-source tool that implements the SQALE method for evaluating technical debt which is currently the most widely used approach to manage technical debt [6].

9.6 Related Work

Past literatures have been focusing mainly on code smells and their impact on software quality [10, 11]. Our work looks into numerous aspects of software system and the correlation each of them has with technical debt and technical debt density.

Marinescu [12] proposed a framework that can assess technical debt by detecting several design flaws in software systems, for example, specific violations of well-established design principles. Zazworka et al. [13] also focused on the impact of design debt on software quality. They investigated how design debt, in the form of god classes, affects the maintainability and the correctness of a software project. However, Sterling [14] mentioned that technical debt does not include only design debt, but also configuration management debt, quality debt, platform experience debt, etc. In this paper, we look at the other different aspects of software system that could potentially relate to technical debt.

Seaman et al. [15] discuss proposals, benefits of managing technical debt, and several decision-making approaches to technical debt. Similarly, our paper aims to help system engineers and decision makers to clearly identify what relates to technical debt and thus supports better management and decision-making with regard to technical debt, again subject to the limitation of our results to open-source software development.

9.7 Conclusions and Future Work

We employed various statistical methods to investigate how technical debt and technical debt density relate to different system characteristics and process characteristics across a representative sample of 91 Apache Java open-source projects. From the results of the data analysis on the hypotheses, we can conclude for similar systems that the size of a software system and the software domain it belongs to can correlate with its technical debt and technical debt density significantly. While the number of system releases and commits has a significant positive relationship with its technical debt, the results show no significant relationship between system technical debt and the number of its contributors and branches. In addition, our analyses show no significant relations between any of these system process factors and technical debt density.

In the future, we will further the study to understand the reasons behind these relations by conducting a biopsy analysis on these systems. Our ultimate goal is to provide guidelines for decision makers to help them study the trade space by providing what factors introduce more technical debt to the system and the quality per capita in the systems. We will also continue to search for similar sources of technical debt data outside the open-source community.

Acknowledgments This material is based upon work supported in part by the U.S. Department of Defense through the Systems Engineering Research Center (SERC) under Contract H98230-08-D-0171. SERC is a federally funded University Affiliated Research Center managed by Stevens Institute of Technology. It is also supported by the National Science Foundation grant CMMI-1408909, developing a constructive logic-based theory of value-based systems engineering. We also acknowledge the support of National Natural Science Foundation of China No. 91318301, 61432001, 91218302.

References

1. Cunningham W (1993) The wycash portfolio management system. ACM SIGPLAN OOPS Messenger 4(2):29–30
2. Falessi D, Kruchten P, Nord RL, Ozkaya I (2014) Technical debt at the crossroads of research and practice: report on the fifth international workshop on managing technical debt. ACM SIGSOFT Software Eng Notes 39(2):31–33
3. Kruchten P, Nord RL, Ozkaya I (2012) Technical debt: from metaphor to theory and practice. IEEE Softw 29(6)
4. Curtis B, Sappidi J, Szynkarski A (2012) Estimating the principal of an application's technical debt. IEEE software 29(6):34–42
5. Falessi D, Reichel A (2015) Towards an open-source tool for measuring and visualizing the interest of technical debt. In Managing Technical Debt (MTD), 2015 I.E. 7th International Workshop on IEEE, pp 1–8
6. Letouzey J-L, Ilkiewicz M (2012) Managing technical debt with the SQALE method. IEEE Soft 29(6):44–51
7. Jiang Z, Naudé P, Jiang B (2007) The effects of software size on development effort and software quality. Int J Comput Inform Sci Eng 1(4):230–234

8. Welch B (1951) On the comparison of several mean values: an alternative approach. Biometrika 38(3/4):330–336
9. Jones C (2003) Variations in software development practices. IEEE Softw 20(6):22–27
10. Olbrich S, Cruzes DS, Basili V, Zazworka N (2009) The evolution and impact of code smells: a case study of two open source systems. In Proceedings of the 2009 3rd international symposium on empirical software engineering and measurement. IEEE Computer Society, pp 390–400
11. Zazworka N, Spínola RO, Vetro A, Shull F, Seaman C (2013) A case study on effectively identifying technical debt. In Proceedings of the 17th International Conference on Evaluation and Assessment in Software Engineering ACM, pp 42–47
12. Marinescu R (2012) Assessing technical debt by identifying design flaws in software systems. IBM J Res Dev 56(5):9–1
13. Zazworka N, Shaw MA, Shull F, Seaman C (2011) Investigating the impact of design debt on software quality. In Proceedings of the 2nd Workshop on Managing Technical Debt ACM, pp 17–23
14. Sterling C (2010) Managing software debt: building for inevitable change. Addison-Wesley Professional
15. Seaman C, Guo Y, Izurieta C, Cai Y, Zazworka N, Shull F, Vetrò A (2012) Using technical debt data in decision making: potential decision approaches. In Proceedings of the Third International Workshop on Managing Technical Debt IEEE Press, pp 45–48

Part II
System-of-Systems Integration

Chapter 10
Applying the Cybersecurity Game to a Point-of-Sale System

Andrew J. Turner and Scott Musman

Abstract The objective of this chapter is to describe the application of the cybersecurity game (CSG) to a point-of-sale (PoS) system and the knowledge discovered from these activities. All PoS systems process over 195 billion electronic transactions with a volume of over US $28.8 trillion per year. In 2014, there were 79,790 cybersecurity incidents reported affecting PoS. There exists a pressing need to understand the cost-benefit for cybersecurity risk reduction investments; however, risk reduction investments face resource limitations. CSG was applied to a PoS to address this need. CSG is a methodology and software tool that models the cyber risk of information and communication technology (ICT) systems. CSG produces security portfolios that are Pareto optimal against quantitative cyber risk and investment costs. CSG identifies the set of defensive methods that best reduce cyber risk for any given investment level. The nominal risk score without employing any defensive methods is 8,492,934. The best risk reduction can be achieved using all of the tools at a cost of ~250,000; however, 89% of the risk reduction is achieved by spending only ~16% of the cost. Additionally, two defense methods were found to provide major reductions in risk. The first is to segment the network between PoS systems and the remainder of the merchant's ICT system. The second was to encrypt information throughout the merchant's ICT system.

Keywords Cybersecurity • Risk • Resilience • Game theory • Point of sale • Modeling • Simulation

A.J. Turner (✉)
The MITRE Cooperation, M/S170, 202 Burlington Road, Bedford, MA, USA
e-mail: ajturner@MITRE.org

S. Musman
The MITRE Cooperation, M/S320, 7515 Colshire Dr,, McLean, VA, USA
e-mail: smusman@MITRE.org

A.M. Madni et al. (eds.), *Disciplinary Convergence in Systems Engineering Research*, https://doi.org/10.1007/978-3-319-62217-0_10

10.1 Introduction

Point of sale (PoS) refers to the system of systems (SoS) that enables electronic transactions (e.g., credit, debit, and gift card transactions). The PoS SoS is composed of numerous systems that serve multiple actors each performing their function in providing payment services between a customer and a merchant. The PoS SoS is so commonplace in everyday commerce; it's easy to be oblivious to its size and complexity. Between the customer swiping a card and the merchant receiving authorization, the authorization request passes through 3–5 other actors. The volume of transactions and capital traversing this network is staggering. In 2014, there were an estimated 195.56 billion transactions [1] with a total volume of 28.844 trillion US dollars (USD) [2]. In the 12 months ending December 31, 2014, a total of 60.1 billion transactions and 4.498 trillion USD in purchases were made with MasterCard credit and debit cards [3]. Similarly, in the 12 months ending June 30, 2015, a total of 103.2 billion transactions and a total of 7.390 trillion USD in purchases were made with VISA credit and debit cards [4]. It is projected that there will be 515.42 billion transactions in 2024 [1].

Naturally, the presence of this massive amount of capital has attracted criminals. Cyber attacks on PoS are prolific. Based on the Verizon Data Breach Investigation Report, there were 79,790 security incidents and 2122 confirmed data loss events across 61 countries in 2014 [5]. The average size of a data breach in the United States in 2014 was 28,070 records [6]. In 2014, the global cost of fraud on the PoS SoS was estimated at 16.31 billion USD [2]. The 2014 US cost of fraud on the PoS SoS was estimated at 7.86 billion USD (i.e., 48% of global). This translates into a cost for every 100 USD transaction of 5.65¢ globally and of 12.75¢ for the United States [2].

The attacks on PoS create a pressing need for the contributing actors to invest in cybersecurity and cyber resilience. However, these investments are constrained by the resource limitations of the actors. To analyze the value of cybersecurity investment, we applied the cybersecurity game (CSG) to a PoS system. CSG is a methodology with supporting software that identifies the optimal security portfolios that can minimize an information and communication technology (ICT) system's cyber risk for any given investment level [7]. CSG employs a game formulation using a process model of a system, an attacker model, a system topology model, and a defender model. The game formulation identifies the strategies that minimize the maximum risk (MiniMax) to a system.

The objective of this chapter is to describe the modeling and analysis performed on the PoS SoS using CSG and the knowledge discovered from these activities. For a more detailed description of CSG, see [7]. The next section provides an overview of the PoS SoS. This is followed by a description of the model developed and analysis performed on a portion of PoS SoS. Finally, this chapter concludes with a discussion on the findings of the CSG analysis.

10.2 The Point-of-Sale System of Systems

10.2.1 Point-of-Sale Overview

The PoS SoS can be generalized as containing five main actors (i.e., customer, merchants, acquirers, issuers, and card brand) and two common actors (i.e., payment gateways and payment processors). The customer is the actor that makes the purchase from the merchant's store and physically owns the cards (e.g., credit cards, debit cards, gift cards). The merchant is the actor that physically owns the merchandise, the store, and has the supporting PoS equipment (e.g., the terminals, servers, PoS software). The merchant maintains an account with one or multiple acquirers to process their accepted card brands and payment types. The majority of merchants do not interact directly with an acquirer. Instead, they contract with multiple processors using a gateway. The processor is a subcontractor to one or multiple acquirers. The gateway provides services to the merchant to route transactions to multiple processors. These two actors route the request to the appropriate acquirer.

The acquirer is a bank or financial institution that contracts with processors or merchants to process electronic payments for a card brand (e.g., VISA, MasterCard) and type (e.g., debit card, credit card). The acquirers check with the issuer for available credit or account balance, authorize the payment transaction, and settle with the card issuer. The issuer is a financial institution that holds and maintains the account associated with the customer card. For example, the issuer of a debit card would be the bank for which the account is operated by the customer. The issuing bank issues payments to the acquiring bank on behalf of its customers. In doing this, the issuer assumes liability on the customer's ability to make payments.

10.2.2 Point-of-Sale Activities

At the merchant, payment services can be defined by two main activities: authorization and settlement. Settlement is sometimes decomposed further into the activities of batching, clearing, and funding. Authorization is the process of checking the cardholder's credit or account balance for available funds. The left image of Fig. 10.1 shows the process of authorization. For in-person purchases (here we do not cover card not present purchases), the customer will present their card to a PoS card reader so that the merchant can obtain information about the card holder and specific information about the card. The merchant will send a transaction request that includes the account information and the purchase amount. The request is then routed by the payment gateway and the payment processor to the acquirer. The acquirer checks with the issuer that checks the account for sufficient credit or funds.

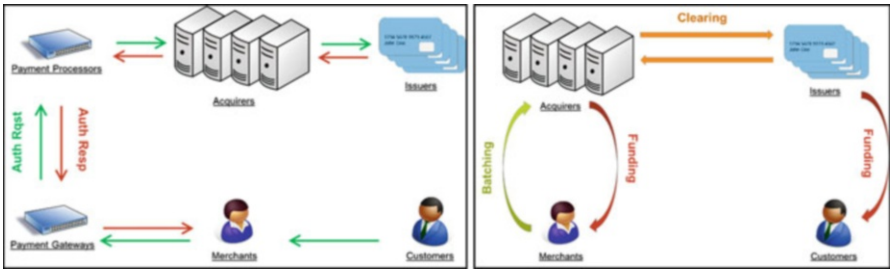


Fig. 10.1 (*Left*) Authorization process. (*Right*) Settlement process

Then, the issuer sends a transaction response with either an authorization or denial of the purchase. The authorization response is then passed through the system back to the merchant. Assuming the transaction is approved, the merchant then gives the customer the merchandise. The approved transaction record is added to the merchant's daily batch. The daily batch is the complete set of approved transaction records that have occurred since a previous batch was reconciled with the acquirer.

Settlement is the process of reconciling the payments between the merchant, its acquirer, the issuer, and the customer. An overview of this process can be seen in the right image of Fig. 10.1. This is often partitioned into three steps: batching, clearing, and funding. In batching, the merchant stores the sales for a given time period in a batch, typically a day. Then the batch is sent to the acquirer to receive payment. During the clearing process, the acquirer sends the received batches to the issuers for requested payment. The issuers then transfer the payment to the acquirers. During the funding process, the acquirers send payment to the merchants and the issuers bill the accounts of the cardholders.

10.2.3 The Modeled System

We modeled a PoS system that was implemented in MITRE's Mobile Computing Security Initiative (MOCSI) lab to investigate cybersecurity aspects of the system. This system is composed of a basic merchant configuration and a test account with a processor. The PoS system executes the authorization process, enabling interested parties to run experiments and assess cybersecurity solutions for the merchant. The implemented system focuses on the merchant; therefore, the roles of the acquirer and the issuer are not included in the CSG model. A basic diagram of the lab setup is shown in Fig. 10.2.

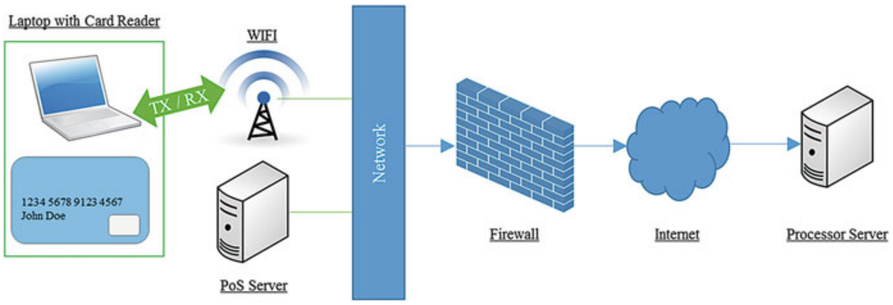


Fig. 10.2 The point-of-sale lab network

10.3 Applying the Cybersecurity Game to Point of Sale

CSG is a methodology and software tool that uses game theory to model and analyze an ICT system with the objective to minimize the system's cyber risk [7]. The optimal set of defensive methods (i.e., an optimal portfolio) that minimize the cybersecurity risk will be different depending on the system usage context and amount of resources one is willing to allocate (e.g., money). As defenders reduce risks in some portions of the system, attackers just shift their focus to other unprotected or less protected portions of the system. To combat this, CSG uses game theory to ensure a balanced defense portfolio and avoid overinvesting in reducing some risks at the expense of underinvesting in defending others. CSG is formulated as a two-person zero-sum game (one where both the attacker and defender assign the same value to gain or loss). As such it implements a rational approach to cybersecurity decision making, where both players play to the best of their ability and work to best counteract each other's moves. CSG optimizes its decision assuming that the attacker knows, or is able to find out, everything about the system they are attacking. CSG focuses on long-term defense employment, where attackers are not necessarily time constrained in their actions. Methods for trying to deceive, delay, or deter an attacker over the short term would motivate a different game formulation. Examples of these different formulations are shown in Roy et al. [8].

CSG is formulated to find the optimal security portfolio, given a performance or cost target. Alternatively, it can perform a more exhaustive search through the set of solutions to find the Pareto frontier across each cost point. CSG uses several different models to produce these results. Its models are a process model that is used to compute impacts, a built-in default attacker model to describe how an attack can move through the ICT system, a system topology model that describes the ICT system component interconnectivity and trust relationships, and a defender model that describes the defensive methods available within the ICT system.

10.3.1 CSG's Quantification of Cyber Resilience and Cyber Risk

CSG produces security portfolios that minimize quantitative cyber risk for any investment cost point. CSG treats cyber resilience as the inverse of cyber risk [9]; therefore, methods that improve an ICT system's cyber resilience result in reducing its operational cyber risk. CSG defines individual risk as the product of the probability that a cyber incident will occur (i.e., P_{CI}) and the expected loss incurred from the incident (i.e., L_{CI}). The loss is some value that accounts for the system's initial degradation and potential recovery over some time horizon due to the cyber incident. CSG then defines a system's total cyber risk as the summation of all the incident risks associated with the set of (mutually exclusive) incidents that an attacker can cause, as shown in Eq. 10.1 [10]. However, total cyber risk can be defined differently. A risk-averse decision maker may choose to reduce the worst-case risk scenario, as shown in Eq. 10.2. Both treatments of system risk are represented in CSG. The measure of total cyber risk used in this chapter is the summation of all cyber risks, as shown in Eq. 10.1:

$$\text{Risk} = \sum_{\text{CI}=1}^{N} P_{\text{CI}} L_{\text{CI}} \tag{10.1}$$

$$\text{Risk} = \max_{\text{CI} \in \{1,\ldots,N\}} P_{\text{CI}} L_{\text{CI}} \tag{10.2}$$

10.3.2 The Cyber Mission Impact Assessment Model

The Cyber Mission Impact Assessment (CMIA) model uses a process model to determine the losses incurred from cyber incidents (i.e., the L_{CI} component from Eq. 10.1). The CMIA software is based on Business Process Modeling Notation specialized for cyber modeling and analysis [11]. The CMIA software makes it possible to capture mission process details as an executable simulation. It models activities, activity durations, activity dependencies, resource dependencies, time constraints, and control flows. This allows the model to capture the varying impacts due to, when an attack occurs, the duration of the attack, the use of redundant systems, and intelligent mitigation strategies. These capabilities allow us to gather a more holistic assessment of losses due to cyber incidents.

Our modeling process for developing CMIA models involves a high-level description of the process, a process model for the incident impacts, and process models for the ICT-dependent activities. For PoS, these models are described below. More details can be found in the MITRE technical report [12]. The development of a CMIA model should start with modeling the high-level process. This is often modeled with a series of submodel objects with control logic and timing.

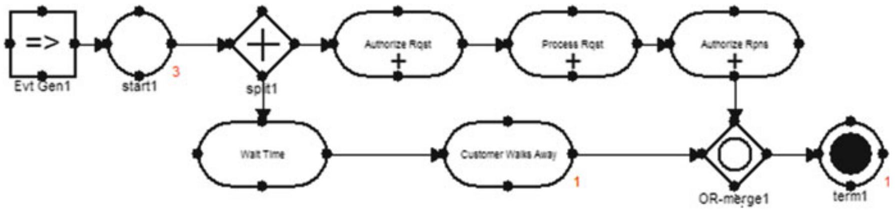


Fig. 10.3 CMIA high-level model for point of sale

A CMIA PoS high-level model was developed for authorization, as shown in Fig. 10.3. It is split into two paths. The top path transitions through the authorization process. First, a transaction request is sent. Then the request goes through processing, resulting in either an authorized or declined purchase. Finally, the authorization response is returned to the merchant. Each of these actions is represented with submodels. If authorization is denied, then the customer and merchant will try again until the time limit expires. If authorization is approved, then the goods are given to the customer and the process ends. The bottom path is a countdown of 10 min until the customer leaves due to frustration and a lost purchase occurs. Therefore, if a cyber attack causes a delay for more than 10 min, then the customer will walk away and the global variable "losses" will be updated.

Next, an impact model is created that includes the defined impact events. The impact events represent an unacceptable outcome to the system as the result of a cyber incident. The impacts are modeled in the CMIA software as a catch event, which can then modify the model variables, the process flow, or both. Note that the impact events that we modeled lead to losses but are not losses themselves. The PoS impact model defines five impacts. These impacts are defined into two categories: loss of customer card data and illegal purchases.

Based on the Verizon Data Breach Investigation Report [5], the financial loss of an individual credit card record is commonly approximated to be 201 USD [5]. This report also provides an estimate for the cost of large data breaches in USD. This is estimated as an exponential model: $\text{Losses} = 3618.2 \times \text{NumRecords}^{0.4236}$. Our model approximates that a merchant data breach would result in 1000 records lost.

The illegal purchases were approximated to average 500 USD each. It is reported that every 1 USD committed in fraud results in 2.79 USD in expenses to the merchant [5]. If we assume organized criminal activity, then this group could average a purchase every 5 min for a 12 h day, resulting in 144 illegal purchases. Thus, on average, we estimate that an illegal purchase results in a 1395 USD loss and multiple illegal purchases result in a 16,740 USD loss.

The last step of the CMIA process is the development of the ITC-level models. These models are developed by defining the ITC activities, their assigned ITC resource, and the interconnected flow between the activities. Example ITC resources are hardware, software, and data. The activities are then linked to the impacts for different cyber incident effects. It is common to consider confidentiality, integrity, and availability (CIA) cyber incident effects, but in CMIA we use the

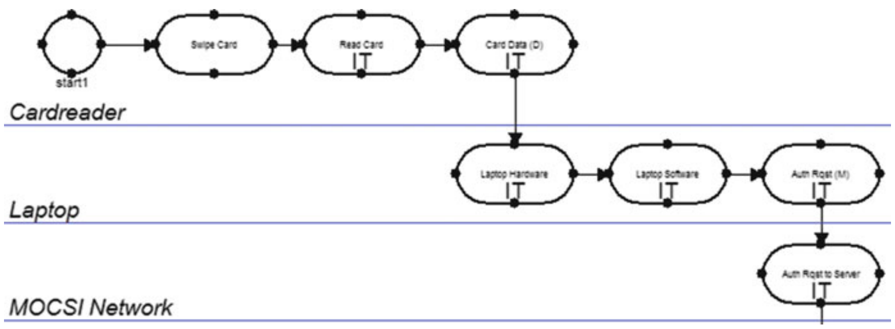


Fig. 10.4 CMIA authorization request ITC-level model

DIMFUI taxonomy as a slightly more comprehensive set of incident effects. DIMFUI represents six cyber incident effects: degradation, interruption, modification, fabrication, unauthorized use, and interception. See Temin and Musman [13] for a detailed discussion on DIMFUI.

A portion of the authorization request submodel is shown in Fig. 10.4. This is a process where the transaction information is created and sent to the processor. The swim lanes break up this process into card reader, the laptop, the MOCSI network, the server, the Internet, and finally the processor activities. Each ICT activity is the performance of an action by the named ICT resource. Impacts are defined for each activity in the event that the resource has been affected by a cyber incident. For example, if the MOCSI firewall is affected by a modification, unauthorized use, or fabrication cyber incident effect, then the impact would be the merchant losing all customer records. Each ITC activity is developed in a similar manner. Details into the other ITC-level models and each supporting ITC activity are omitted for brevity.

10.3.3 *The Attacker Model*

In order for the attacker to affect ITC resources that can cause significant impacts, the attacker must find a pathway to access them. An individual incident may cause no impact on its own but can act as stepping stones for follow-on attacks. Many risk assessment methods [14, 15, 16] either fail to consider these noncritical ICT resources or model them implicitly. CSG provides a default attacker model that defines the probability that attacks will succeed given the topological constraints that the system topology imparts on the attacker. The attacker model specifies two points of entry for an attacker. Attackers can try and enter from the Internet or be a malicious inside user. From either entry point, the attacker can move through the network to reach ITC resources and create cyber incident effects that can cause an impact. The attacker model represents the ability of an attacker to move through the network as a series of attack steps each with a probability of succeeding.

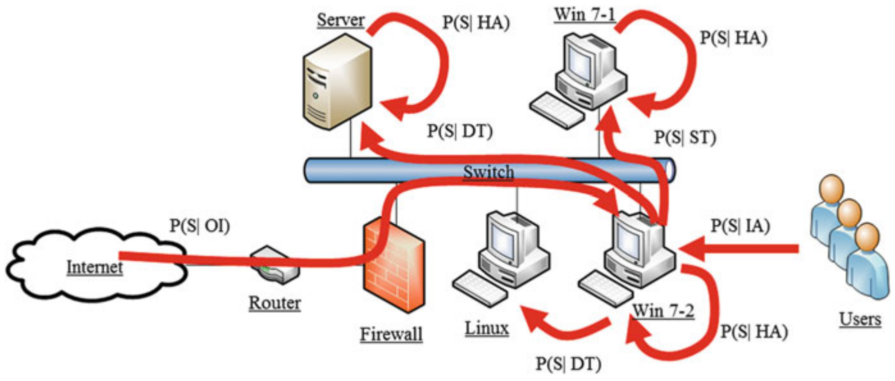


Fig. 10.5 Attacker success probabilities

The attacker model conditions the probability of an attack succeeding with the following characteristics:

- Whether the attacker is trying to compromise a component to which they can directly connect (i.e., inside the network where they reside), or whether it requires crossing a network boundary to access
- Whether that component is the same type as one of the components they have already compromised
- Whether a component is known to be vulnerable to known exploits that a current attacker is likely to possess
- Whether the component is a server that contains one or more network services
- Whether users who fill roles that have access to each resource have the ability to leverage those user roles to access other components that can create impacts

Figure 10.5 shows example probabilities of attacker success for a sample topology, where the client host Win 7-2 is the first target of an attacker. The diagram shows that the client host, Win 7-2, can become successfully compromised by an attacker from outside the network with probability P(S | OIC), where S is a successful compromise and OIC is the attacker's situation Outside trying to get In by attacking a Client (OIC). If the attacked host is a server, then P(S | OIS) would be used. The client host, Win 7-2, can also become compromised by a malicious insider with Inside Access (IA) with probability N * P(S | IA), where the N represents the number of users that have access, and the value does not exceed unity.

Once host Win 7-2 is compromised, the attacker Has Access (HA) and so applications or data that are on that host can be compromised with probability P (S | HA). Because host Win 7-1 is the Same Type of Client (STC) as Win 7-2, the same exploit used to compromise Win 7-2 has a high chance of being able to compromise Win 7-1 with probability P(S | STC). Since the Linux host is a client computer of a Different Type of Client than Win 7-1, it will have a different chance of success, P(S | DTC). Server A is also a different type of host than Win 7-2, but because it is a server, it will have one or more network services that could be exploited. It will use the P(S | DTS) probability. A summary of the attacker model probabilities used are P(S|OIS) = 0.002, P(S|OIC) = 0.001, P(S|IAS) = P(S|IAC) = 0.000001, P(S|STS) = P(S|STC) = 0.9, P(S|DTS) = 0.02, and P(S|DTC) = 0.01.

10.3.4 The System Topology Model

CSG computes a risk score using the impacts from the CMIA model and the probability that the impacts will occur from the attacker model given the constraints of the system topology. The topology model represents the interconnection of ITC resources in the system. These resources include ITC components, applications, data, user account groups, and firewalls or access controls that implement trust relationships. Additionally, items in the topology model include single ITC resources, as well as ITC resource pools that represent functionally identical groupings of resources of the same type. The system topology model requires resource type information for each ITC resource, so CSG knows whether the same attacker exploit from an earlier step can be used against them or not. The existence of connections, firewall rules, and the access of user roles define connectivity capabilities and restrictions between ITC resources. Figure 10.6 illustrates the PoS topology model that matches up to the network diagram shown in Fig. 10.2 and the ITC activity models shown in Fig. 10.4. The PoS topology model captures hosts, applications, and data along with where they sit on the network, their component type, and connectivity trust relationships; even though these last two are not shown in the figure.

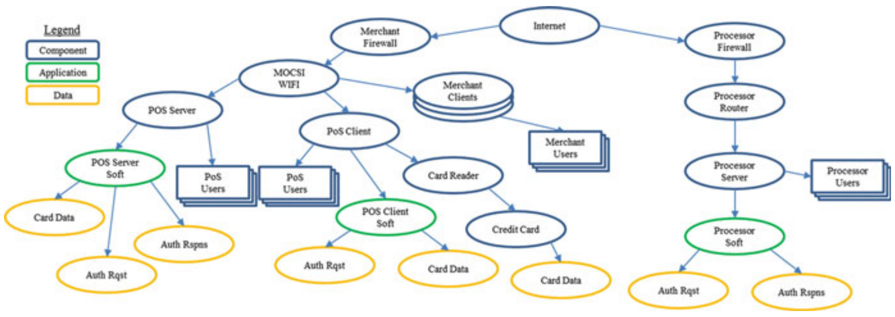


Fig. 10.6 PoS topology model

10.3.5 The Defender Model

The Defender model is a collection of defensive methods that can be applied to the system to reduce the cyber risk. A defender can change the configuration of the systems topology, which will make it harder for attackers to access critical resources. A defender can deploy tools that can make it less likely that attacks will succeed (e.g., host hardening). Finally, a defender can deploy tools that reduce the impacts of successful attacks (e.g., redundant spares). The set of defensive tools considered for the PoS system are listed in Table 10.1.

The defender model includes the set of security methods recommended by cybersecurity engineers. These include several forms of access control: encryption; configuration management of the PoS server and clients; Network Access Controls (NAC) to limit unauthorized connections to the network; Network Intrusion Detection System (NIDS) which can include application monitoring of the PoS service on the server; file integrity checking on the PoS server; Host-based Intrusion Detection System of the clients and servers; a solution for periodic re-imaging of the PoS clients to reset them to a known safe state; white listing of processes on the PoS components; the use of the EMV-standard chip technology; and a method called tokenization which locally stores customer and card details as tokens that do not contain the card or customer information.

Each row of the table describes a different tool that employs a defensive method. Tools that belong to the same category are applied exclusively of other tools in that category. For example, only one of the three permutations of Terminal Access Control will be used in a portfolio. The total cost of employing each tool is the summation of install, maintenance, and operation costs. The method effectiveness assessments define how well the method works to mitigate each cyber effect. These values were obtained from SME interviews. The effectiveness assessments are defined as a 0–100 score. For example, a value of 40 implies that 40% of the attacker exploits that we know about or anticipate would fail. Finally, each row contains a list of the specific ITC resources they are applied to.

In addition to the defined defensive tools shown in Table 10.1, a PoS topology variation with implemented diversity was assessed. A second PoS topology model was developed that segments the PoS resources from any other merchant ITC resources that are not related to PoS. This topology variation also specifies diversity of the ITC resources used for the merchant PoS clients and the PoS server, so the same exploit cannot be used to subvert both resources.

10.3.6 Results

Using the defensive methods described previously, we used CSG to perform a portfolio analysis of all 55,295 defensive combinations. These results are shown in Fig. 10.7. Additionally, a sample of the security portfolio results are shown in

Table 10.1 PoS defensive tools

Category	Method	Applies-to	Tool	Install cost	Maintenance cost	Operational cost	Total cost	Interruption	Modification	Fabrication	Unauthorized-use	Interception
Terminal Access	Passwords/ Token	POS Server Software ...	Access Card	10000	500	2000	12500	0	40	40	50	50
Control												
Terminal Access Control	Passwords/ Autolock / Logout	POSServer Software ...	Built-in +manual config	750	50	3000	3800	0	20	20	40	40
Terminal Access Control	Passwords/ Token/A utolock/ Logout	POS Server Software ...	Access Card & DB module	10500	550	5000	16050	0	40	40	60	60
Encryption	Disk & trans-port Encryption	Cust Accnt Reds ...	LUKS&TLS	1000	0	50	1050	0	40	40	40	60
Server Con-figuration Management	Harden Server	POS Server Software ...	Puppet	2000	2000	100	4100	20	20	20	20	20
POSterminal Configuration Management	Harden POS Term	Laptop POS Software .	. MaaS360	55000	5000	1000	61000	20	60	20	40	20
Network Access Control	NAC	MOCSIWifi...	Cisco ISE	30000	3000	1000	34000	0	20	20	60	20

Network Access Control	NAC + VPN	MOCSIWifi...	Cisco ISE +	35000	3000	-5000	33000	0	20	20	60	20
Detection	NIDS	POS Server Hardware ..	Security Onion	5000	500	0	5500	20	20	20	20	20
Network Intrusion Detection	NIDS +Applications Monitoring	POS Server Software ...	ModSecurity	10000	0	0	10000	20	20	20	20	40
Server Intru-sion Detection	File Integrity	POS Server Software ...	Tripwire	15000	1500	1000	17500	0	40	20	20	20
T okenize	T okenize Transactions	Cust Accnt Reds ...	First Data	30000	1000	0	31000	0	0	0	90	90
Host Intrusion Detection	Virus detec-tion/HIPS	Laptop POS Software. . .	Semantec Endpoint Protection	5000	500	1000	6500	20	20	20	20	20
EMV	CHIP n Sg	CardData	PCI	20000	1000	500	21500	0	0	60	0	0
EMV	CHIP n PIN	CardData	PCI	20000	1000	10000	31000	0	0	85	60	0
ReimagePOS Systems	Periodic POS terminal re-image	Laptop POS Software . . .	N/A	25000	1000	5000	31000	20	60	20	50	20
Whitelist processes	White Listing	POS Server Software ...	Bit9 Parity	5000	0	500	5500	40	60	20	60	25

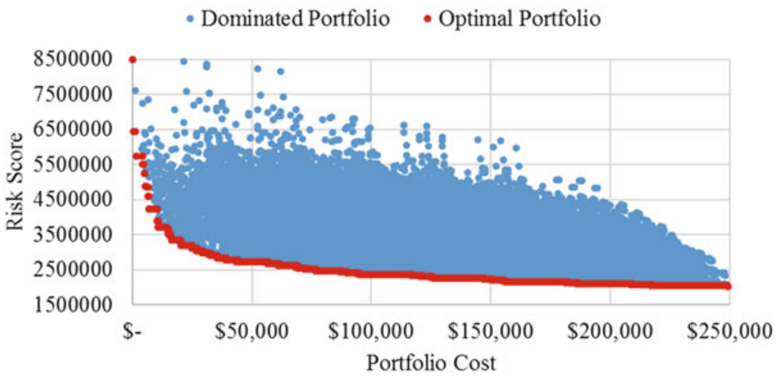


Fig. 10.7 PoS portfolio analysis results

Table 10.2 Select security portfolios (superscript P reflects a Pareto optimal portfolio)

Cost rank	Portfolio defenses	Cost	Risk
1^P	Segment	500	6,441,843
3^P	Disk &amp; transport encryption + segment	1050	7,613,147
10^P	Passwords/autolock/logout + disk &amp; transport encryption	4850	5,238,869
38^P	Passwords/autolock/logout + disk &amp; transport encryption + white listing	10,350	3,892,103
570	Segmentation + passwords/token + NIDS + applications monitoring + virus detection/HIPS + white listing	35,000	3,418,609
581	Harden server + CHIP n PIN	35,100	7,114,348
55295^P	Segmentation + passwords/token/autolock/logout + disk &amp; transport encryption + harden server + harden POS term + NAC + NIDS + applications monitoring + file integrity + tokenize transactions + virus detection/HIPS + CHIP n PIN + periodic POS terminal re-image + white listing	249,200	2,038,408

Table 10.2, ranked by cost. A superscript P is used on the cost rank to indicate a Pareto optimal portfolio.

As can be seen in Fig. 10.7, different combinations of defense methods can produce radically different amounts of risk deduction at any price point, where the risk score here represents an expected loss in USD. The portfolios ranked between 570 and 581 all require the same cost (around 35,000 USD) but provide a 3.7 mil USD range in risk value. This is why only the Pareto optimal portfolios are considered for selection, but selecting an optimal portfolio is not the only concern.

The nominal risk score without employing any defensive methods is 8,492,934. The best risk reduction can be achieved using all of the tools at a cost of ~250,000; however, 89% of the risk reduction is achieved by spending only ~16% of the cost. We can get a better understanding of which defensive methods need to be invested in by looking at commonly occurring tools in the optimal portfolios. Segmenting the network and encrypting the information shows up in almost every Pareto optimal portfolio. Both are cheap defensive methods. The first makes it harder for an attacker to access the cyber resources that cause impacts. The second reduces the probability that a successful attack would enable an attacker to interpret the contents of data they obtain.

10.4 Conclusions

This chapter detailed the application of the cybersecurity game to analyze a PoS system that was implemented in a MITRE lab. This system is composed of a basic merchant configuration and a test account with a processor. CSG applies a game theoretic approach to implementing a comprehensive, quantitative assessment of a mission system's cyber risk and reduction of those risks. By applying game theory, the risks are reduced systematically. Each defender move attempts to protect the components associated with the largest risks existing at that point, which may then cause some other component(s) to then have the largest risk. Therefore, CSG portfolios are balanced to reduce overall risk. CSG accomplishes this by employing a process model of a system, an attacker model, a system topology model, and a defender model.

CSG found that the best risk reduction can be achieved using all of the tools at a cost of ~250,000; however, 89% of the risk reduction is achieved by spending only ~16% of the cost. Additionally, two defense methods were found to provide major reductions in risk. The first is to segment the network between PoS systems and the remainder of the merchant's ICT system. The second was to encrypt the information throughout the merchant's ICT system.

Disclaimer The authors' affiliation with The MITRE Corporation is provided for identification purposes only and is not intended to convey or imply MITRE's concurrence with, or support for, the positions, opinions, or viewpoints expressed by the authors. 16-3460

References

1. The Nilson Report, Charts & Graphs Archive, 2016. [Online]. Available: https://www.nilsonreport.com/publication_chart_and_graphs_archive.php?1=1&year=2016#. Accessed 25 March 2016
2. MarketWatch, "Global Card Fraud Losses Reach $16.31 Billion — Will Exceed $35 Billion in 2020 According to The Nilson Report," 4 Aug 2015. [Online]. Available: http://www.marketwatch.com/story/global-card-fraud-losses-reach-1631-billion-will-exceed-35-billion-in-2020-according-to-the-nilson-report-2015-08-04. Accessed 25 March 2016
3. MasterCard (2015) Supplemental operational performance 2015Q2– 2013Q1. MasterCard
4. VISA (2015) Operational performance data 2Q2015. VISA
5. Verizon (2015) 2015 data breach investigation report. Verizon
6. Ponemon Institute, (2015) 2015 cost of data breach study: global analysis. Ponemon Institute

7. Musman S (2015) Playing the cyber security game: a rational approach to cyber security and resilience decision making. MITRE Technical Report, MTR-150371, McLean, VA
8. Roy S, Ellis C, Shiva S, Dasgupta D, Shandilya V, Wu Q (2010) A survey of game theory as applied to network security. In 43rd Hawaii International Conference on System Sciences (HICSS), Koloa, HI
9. Musman S, Agbolosu-Amison S (2014) A measurable definition of resiliency using "mission risk" as a metric. The MITRE Corporation, McLean, VA
10. Clemens PL, Swallom DW (2005) Summing risk — an international workshop and its results. 41(6)
11. Musman S, Temin A (2015) A cyber mission impact assessment tool. In IEEE International Symposium on Technologies for Homeland Security, Waltham, MA
12. Turner AJ, Musman S (2016) Using cyber impacts to assess cyber risk mitigation on point of sale systems using CMIA and CSG. MITRE Technical Report, MTR-160339, Bedford, MA
13. Temin A, Musman S (2010) A language for capturing cyber impact effects. MITRE Technical Report MTR-10344. MITRE Corporation, Washington, DC
14. Garvey PR, Patel SH (2014) Analytical frameworks to assess the effectiveness and economic-returns of cybersecurity investments. In Military Communications Conference (MILCOM), 2014 IEEE, Baltimore, MD
15. Carin L, Cybenko G, Hughes J (2008) Cybersecurity strategies: the QuERIES methodology. Computer 41(8):20–26
16. Buckshaw D, Parnell G, Unkenholz W, Parks D, Wallner J, Saydjari O (2005) Mission oriented risk and design analysis. 2

Chapter 11
Resilient Cyber-Secure Systems and System of Systems: Implications for the Department of Defense

Wendy Leonard

Abstract Over the past two decades, the United States has continued to modify and improve upon its cyber strategy as a result of a constantly evolving and asymmetric cyber threat. As the dependency on networked systems and connectivity has increased, so have the complexity and vulnerability of these systems within US critical infrastructure. For this reason, resilience, affordability, and collaboration between the government and private industry will be imperative in maintaining the cyberspace advantage as cyber threats groups continue to target systems and system-of-systems (SoS) within the Department of Defense (DoD) (Syst Eng 15:95–107; Wheaton MJ (2016) Affordable resilient systems. Engineered resilient systems and system-of-systems. University of Southern California, Olin Hall, Los Angeles, 4 Apr 2016, Lecture). Systems engineering concepts such as trade-space analysis and systems thinking, in concert with an emphasis on resilience at critical system nodes and boundaries, can help reduce system vulnerability when confronted by a constantly adapting cyber landscape. Given the perpetual and rapid evolution of cyber threats due to technological advances and network reliance, designing critical infrastructure systems for survivability is no longer sufficient. This paper identifies current limitations in the nation's cyber strategy and recommends approaches to fill those gaps.

Keywords Resilient • Resilience • Department of Defense • Cybersecurity • Cyber threat • Systems • System-of-systems

11.1 Introduction

Over 20 years ago, on July 15, 1996, President Clinton published Executive Order 13010, which warned of not only physical threats to critical infrastructure but also newly dubbed "cyber threats" that could debilitate US defenses [1]. Critical

W. Leonard (✉)
University of Southern, Los Angeles, CA, USA
e-mail: wendy.leonard@us.af.mil

A.M. Madni et al. (eds.), *Disciplinary Convergence in Systems Engineering Research*, https://doi.org/10.1007/978-3-319-62217-0_11

infrastructure is defined as physical or virtual systems that if incapacitated or destroyed would lead to debilitating impact on some form of national security or public health and safety [2]. Critical infrastructure is a high-value and often highly vulnerable target for many of our adversaries, particularly due to the expanding nature of the cyber realm to an "online battlefield," where data are so freely relinquished by users, leading to illicit exploitation by not only terrorist organizations but also nation-states and criminal enterprises [3].

The cybersecurity landscape has continued to evolve at an alarming rate and sophistication over the last two decades, including attacks by both state and non-state actors [4]. As a result, the US Department of Defense (DoD) developed its first cyber strategy in 2011 and has steadily worked toward developing more survivable cyber-secure systems and system-of-systems (SoS) [4]. The current strategy promotes an anticipatory, dynamic, flexible, and agile environment with collaborative partnerships at its core. Resilience can be achieved by incorporating not only the ability to anticipate emerging threats, but also the ability to resist and absorb cyber attacks without degradation in performance and ability to continue the mission, and adapt and recover from a cyber attack through reconfiguration and restoration [5–7]. The DoD cyber-secure system is a complex system because of its vast number of subsystems, interconnections and dependencies between those subsystems, and human operators working within the system. Human roles include both IT professionals and end users. The need for resilience puts an extra layer of complexity on the system [8]. This complex cyber-secure system exhibits emergent behavior through unpredictable outputs due to unintended and likely innocent interactions with cyber threat actors.

These cyber threat actors do not discriminate between the private industry and government sectors. This is evidenced by the numerous attacks on health care and financial institutions across the world, as well as the 30 million malicious attacks against DoD systems between September 2014 and June 2015 – of which approximately 30,000 were successful [9, 10]. Furthermore, cyber attacks stem from a multitude of threat actors, including nation-states such as China, Russia, Iran, North Korea, and Syria, and non-nation-state threats such as so-called hacktivists and patriotic hackers [11]. According to some sources, the most sophisticated cyber attacks are carried out by the aforementioned groups rather than actual terrorist organizations such as the Islamic State or their affiliates [11]. This does not mean, however, that terrorist organizations will not someday use cyber crime to meet their objectives by attacking US military forces through the DoD. In this case, for threats who do not possess the technical capabilities to create cyber attack tools, it is conceivable that they would recruit skilled individuals and use publicly available tools to carry out spear phishing routines for example, or use social engineering to collect information through malware, which could ultimately be used to impede DoD mission objectives [11]. Furthermore, these terrorist organizations could conceivably combine kinetic attacks with unsophisticated, yet effective, cyber attacks such as destruction of data/resources or denial of service [12]. When considering how highly critical data and the timeliness of that data transmission is, when carrying out military operations, these types of joint kinetic and cyber attacks could likely result in devastating consequences. Many once-limited threat organizations with respect to cyber capabilities now have an asymmetric advantage by being able to disproportionately wreak havoc on their enemy as compared to the investment of resources needed to carry out an attack [11].

The DoD is charged with the mission of equipping our military forces with the necessary means to not only deter war but also protect the security of our nation, including its critical infrastructure [12]. More specifically, the DoD's three primary cyber missions are to, "defend DoD networks, systems, and information; defend the U.S. homeland and U.S. national interests against cyber attacks of significant consequence; and provide cyber support to military operational and contingency plans" [13].

According to the testimony by Mr. James R. Clapper, Director of National Intelligence, for the Worldwide Threat Assessment of the US Intelligence Community, there are two key cybersecurity challenges that we face: "The difficulty of providing timely, actionable warning of cyber threats and incidents, such as identifying past or present security breaches, definitively attributing them, and accurately distinguishing between cyber espionage intrusions and potentially disruptive cyber attacks; and the highly complex vulnerabilities associated with the IT supply chain for US networks" [14, 15].

As a result of these challenges, in 2015 the DoD released an updated cyber strategy that stresses deterrence will be key to the future success against cyber attacks [4]. As discussed later in this paper, there must be more than just a focus on effective protection and denial capabilities in order to combat cyber threats, but also a more prominent focus on embracing every aspect of resilience within the DoD cyber systems and SoS in order to withstand the cyber attacks that continue to penetrate US systems.

This paper is organized as follows. Section 11.2 discusses DoD cybersecurity strategy. Section 11.3 discusses limitations and gaps in current cybersecurity methods. Section 11.4 discusses potential approaches to fill current gaps. Section 11.5 discusses practical steps to overcoming existing cybersecurity limitations. Section 11.6 discusses the necessary shift from survivable to resilient cyber-secure DoD systems. Section 11.7 provides a short summary to this paper.

11.2 DoD Cybersecurity Strategy

In 2013, the DoD Chief Information Officer recognized the need to introduce resilience into the DoD cybersecurity strategy in order to combat cyber attacks by focusing its strategic efforts on a "resilient cyber defense posture" through a transformation of its defensive operations and improvement in both situational awareness and survivability of and against cyber attacks [16, 17]. Furthermore, the DoD has outlined extensive policies and guidance to manage cybersecurity threats. These threats continue to plague Government systems through novel techniques and exploitation of system vulnerabilities. Most notable is the

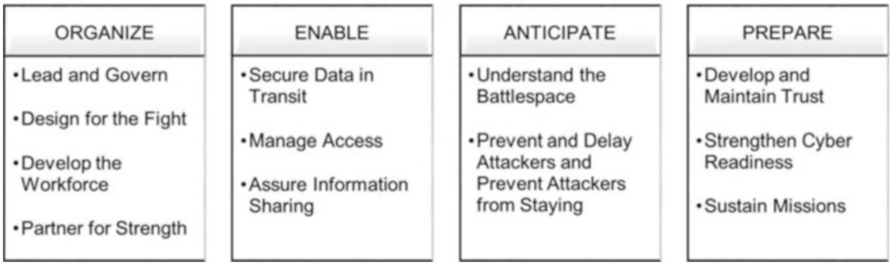

ORGANIZE
•Lead and Govern
•Design for the Fight
•Develop the Workforce
•Partner for Strength
ENABLE
•Secure Data in Transit
•Manage Access
•Assure Information Sharing
ANTICIPATE
•Understand the Battlespace
•Prevent and Delay Attackers and Prevent Attackers from Staying
PREPARE
•Develop and Maintain Trust
•Strengthen Cyber Readiness
•Sustain Missions

Fig. 11.1 Goals for DoD cybersecurity-related policies and issuances [15]

unprecedented and large-scale breach of sensitive personal data in security documents from the Office of Personnel Management in 2015 [9]. In this same year, the DoD organized the aforementioned policies and guidance into the set of goals with an overarching focus on survivability. Figure 11.1 shows these policies and guidance [7].

Cyber attacks continue to grow and cause increasing concern. According to the FireEye system, which tracks threat groups, in February 2016 there were over 500 threat groups actively preparing for and/or engaging in cyber attacks, 29 of which were suspected to be supported by governments outside the United States [18]. FireEye is a system that can identify and analyze threat data from around the world in an effort to better understand adversarial operations in all phases of a cyber attack. This holistic understanding of a threat is necessary when creating a resilient system or SoS with the ability to combat current and emerging cyber threats.

In addition to a holistic understanding of the cyber threat, DoD has recognized that a holistic approach to combating the threat is necessary and can be achieved through a united front between government and the private industry. Under the current cybersecurity strategy, the DoD strives to establish these types of partnerships or command and control relationships using a military strategic framework in order to achieve cyberspace superiority [17]. The strategy further insists upon a condensed decision-making cycle when responding to a cyberthreat as well as fully integrated, deconflicted, and synchronized cyberspace operations to promote collaboration between relevant stakeholders [17]. In recent years, DoD organizations have created partnerships with private-sector companies such as the cooperative research and development agreement (CRADA) between the Air Force Research Laboratory and Rsignia Inc. to create enhanced cybersecurity software to meet the current and future requirements of our warfighter [16]. This partnership is expected to result in innovative cyber capabilities to proactively defend against the next generation of cyber threats facing DoD data and networks [19]. Through a multidisciplinary approach (e.g., combination of engineering, physical science, social science, and computer science), Rsignia plans to leverage on behavioral analysis and modeled intelligence to characterize the future of the cybersecurity domain [19].

11.3 Limitations in Existing Cybersecurity Methods

The current DoD cyber strategy primarily employs a fault-tolerant mindset in which potential threats within a system are assessed and mitigated using a mechanism that can detect a failure following a cyber attack and subsequently attempt to prevent further damage to that system. DoD cyber-secure systems and SoS continue to fall short in securing critical data and infrastructure since the principle focus is not on how to identify, manage, detect, and isolate potential faults prior to a cyber attack, but rather after the attack has occurred. In addition to developing anticipatory and flexible systems that can withstand an attack, system developers must imbue an outward-looking mindset that takes what we know as the current threat and also embraces the inevitability that unexpected forms of cyber attacks will continue to occur until a point at which it is no longer cost-effective for our adversaries. This outward-looking mindset adds to the currently inward-looking focus of combating cyber threats through systems that can avoid and withstand these known threats to develop resilient systems that also adapt to and recover from cyber attacks – yielding a holistic focus and approach to combating the cyber threat.

The DoD currently employs various techniques to defend against existing cyber threats. However, defensive techniques such as the "fortress" approach, which can come at great cost, have proven infeasible to execute and therefore leave our networks unsecure [11]. We must embrace the notion that these cyber attacks will continue to occur at an increasing rate and technical evolution and therefore introduce resilience into DoD cyber systems using a strategy that allows for adaptation and recoverability to account for entire cyber attack life cycle [18, 11]. There are two main areas spanning this life cycle that should be addressed within the DoD in order to contend with the future of cyberthreats and protect our critical information infrastructure. These include awareness and education by the user of network and data management best practices in parallel with improving retention of a skilled cyber workforce within the DoD, and strengthening partnerships and timely information sharing between public- and private-sector organizations [10, 20].

As a result of EO 13010, an Infrastructure Protection Task Force (IPTF) was established to facilitate government and private industry collaboration. While the term "resilience" was not expressly stated, the strategy for protecting critical infrastructure was intended to address some aspects of resilience through threat detection, prevention, elimination, confinement, and recovery/restoration of a system following an attack [1].

Cyber threats against critical infrastructure are one of the greatest challenges for national security and require the establishment of an information-sharing and collaborative partnership between relevant stakeholders in order to improve the volume, timeliness, and quality of shared cyber threat information [2]. Clearly, the measures that have been in place to protect the US critical infrastructure from cyber intrusions over the last two decades are lacking and at a minimum must undergo a shift in strategy mindset and application, particularly to one of complete resilience, in order to be able to more effectively combat the constantly evolving cyber threat. Adherence to DoD policies and guidance may strengthen the systems, but they lack the necessary focus on two main components required to develop a fully resilient cyber-secure solution that can truly address the emergent behavior of cyber threats. In addition to the already addressed need for anticipation and absorption of cyber attacks, a resilient system is one that also has the ability to adapt to and recover from an attack in order to fulfill its originally intended mission [6]. The primary focus of system development must shift due to the elasticity required for mission sustainment and prevalent nature of cyber threats that only continue to evolve and thwart our continued efforts to defend DoD systems.

The EO 13010 included requirements to institute training and education programs with the intent to reduce vulnerabilities as well as respond to attacks targeting US critical infrastructure [1]. According to a memo from the Office of the Secretary of Defense in 2015, poor network architecture, subpar discipline in network and data management, and poor user practices resulted in vulnerable systems and can be directly traced to about 80% of cyber incidents [10]. Moreover, these systems continue to be left open to seemingly ubiquitous vulnerabilities in an environment of networked systems of increasing complexity due to flawed system architectures and development, as they are rushed through testing and fielding with inadequate security requirements employed [3]. As evidenced by the large number of successful cyber attacks on the United States alone, awareness of ongoing cyber developments and threats across the nation is lacking and must be addressed [21].

Coupled with a lack in education and training on cyber threats, the DoD in particular has a difficult time retaining a skilled workforce due to the many enticing factors of the private industry [21]. This results in a lack of technical ability to create and maintain a resilient cybersecurity system. Through partnership between organizations, the effects of poor retention can be mitigated. This lack in current practices lends itself to the idea that the existing mindset used to develop the US cyber strategy is deficient and must be transformed into one that stresses all facets of resilient cyber-secure systems that can account for many of the aforementioned shortfalls.

11.4 Approaches for Curtailing Cybersecurity Shortfalls

Systems engineering is a process that looks at the entire life cycle of a system or SoS. It does not just address one aspect or component, but rather takes into account each piece of a system from a holistic viewpoint. In this manner, the systems engineering mindset can be applied to the problem of cybersecurity that plagues our nation as well as nations around the world in order to minimize the number of successful cyber attacks. The limitations of current practices can be mitigated by developing a strategy that accounts for the entire life cycle of a system from conception through testing, training, operation, and maintenance and also integrates all relevant components and stakeholders throughout the life cycle. A strategy for

Cybersecurity in the Defense Acquisition System was published in January 2017 through a directive-type memorandum (DTM) from the Under Secretary of Defense of Acquisition, Technology, and Logistics. Regardless of the systems engineering methodology applied to a DoD system, implementing Cybersecurity across the entire life cycle must apply the concepts discussed in this DTM prior to material development decisions, during the Materiel Solutions Analysis (MSA) phase, Technology Maturation and Risk Reduction (TMRR) phase, Engineering and Manufacturing Development (EMD) phase, Production and Deployment Phase, and the Operations and Support phase. The effectiveness of this strategy will depend greatly on its ability to apply all four aspects of resilience to the holistic cyber-secure system – the ability to avoid, withstand, adapt to, and recover from disturbances [5]. Aspects to include in this strategy are discussed in Sect. 11.5. System designers must embrace the reality that failure of complex cyber systems will undoubtedly occur and cannot be fully preventable due to existing vulnerabilities as well as those that will be created as technology improves and the nature of networking systems grows. When designing a resilient cyber-secure system or SoS, a resilient cyber-secure strategy must be in place that reduces the intensity of impact following a cyber attack such that the intended mission objectives can continue to be met following adaptation and recovery [22].

When designing this resilient cyber-secure strategy, the concept of systems thinking should also be employed in order to promote an understanding of the big picture and the long-term consequences of that strategy [23]. At the foundation of systems thinking is the ability to operate using a systems-level perspective to include the entire user experience. The types of thinking necessary when designing a cyber-secure system or strategy include: strategic, critical, associative, holistic, and interdisciplinary thinking [8, 23]. In addition to these types of thinking, questions such as when and what type of redundancy to apply (physical vs. functional), how to mitigate the impact of a cyber threat and assure continuity of service, and what level of flexibility is required in the system must be addressed [8, 23].

It is important to recognize that as changes are made to the cyber-secure system to meet emerging threats, these capabilities will increase the system brittleness through additional interdependencies and, thus, complexity [7]. In order to combat this brittleness, the value of each quality attribute as it supports system resilience must be determined. In doing this, factors such as the operational environment, including how users might change or adapt over time, must be considered by the key stakeholders during this process. For this reason, all relevant stakeholders in the cyber community, including both the US government and private industries have typically been involved in the development of the current cyber defense strategy. Trade-offs need to be made as interests in quality attributes and the level of affordability by stakeholders may vary and lead to conflict.

In addition to designing a resilient cyber-secure system from concept initiation, the nodes and boundaries of that system or SoS where independent systems intersect and become interoperable must be evaluated to determine which are most critical and therefore must be afforded the greatest levels of protection

[8]. These nodes and boundaries within DoD SoS will be the most challenging to forecast system behavior and therefore result in vulnerabilities that threat actors will unfortunately exploit. In a networked SoS, multiple nodes can be compromised; however, the SoS might still be able to perform its intended objectives. This means that those nodes were not critical to the mission. However, failure in one critical node can result in catastrophic consequences [18, 23]. For example, consider the DoD SoS that includes critical infrastructure or government entities and contractors/subcontractors; each of these systems or partnering organizations could be considered a node or "threat point" where data are transmitted and shared without a clear understanding of the safeguards in place and often an overreliance on security protocols without due diligence by the parent organization [20]. If a cyber attack is successful and a critical node is compromised, leading to some type of disruption or failure, there could be catastrophic consequences that could put mission objectives at risk.

11.5 Practical Steps to Overcoming Existing Cybersecurity Limitations

Resilient cyber-secure systems in the DoD will be able to deal with cyber attacks and recover from disruption. There will always be numerous tactics used by cyber threats groups. Tactics such as zero-day exploits and spear phishing are most commonly used to gain access to sensitive military and political information. While the consequences of these tactics can be better managed through awareness of existing threat capabilities and training on how to combat an attack, this alone will not be enough, given that these tactics tend to leverage on speed and scale to attack networks. This is where relevant and timely knowledge sharing can amplify the resilience of DoD systems through rapid and adaptable responses necessary for continued operation toward mission objectives.

The first practical step is training of the cyber workforce not only with the intent to combat cyber threats postattack, but to develop awareness in the cyber workforce and among system designers such that systems will be developed that can prevent attacks using well-known tactics from occurring in the first place. Just as architecting a system correctly in the beginning can mean a significantly stronger and more resilient system, not to mention greater affordability in the long term, developing a cybersecurity protocol that focuses on training and compliance, the percentage of successful cyber attack can be decreased. As the cyber workforce becomes more aware of cyber threat tactics, assuming that workforce retention continues to be an issue, it will be imperative that the DoD exploit the best practices from other government agencies and the private industry.

The development of a united front and information sharing between the DoD and other agencies is the second step in developing a more resilient cyber-secure system and SoS. While information sharing between these industries is critical to DoD cyber defense, it will again not fully eliminate cyber attacks. This is where integrating resilience with current fault-tolerance methods in the form of predictive (threat motivations, plans, and intentions) and reactive (technical indicators) tactics becomes essential [9]. As previously mentioned, the timeliness of data transmission pre- and post-cyber attack is vital to carrying out successful defensive and offensive cyber operations. In the same manner, information sharing between stakeholders must also be timely in order to maximize the relevance of that data. While the idea of knowledge sharing between government and the private industry is not a novel one, the widespread implementation, possibly even on a global scale, has not been highly successful, leading to continued penetration by cyber threat groups at the detriment to our nation's critical information infrastructure that serves as the communications and/or services whose quality attributes such as availability, reliability, and resilience are vital to maintaining a secure network of DoD systems [21].

During testimony before Congress on emerging cyber threats, private industry companies such as FireEye discussed their current operations and provided recommendations for strengthening our nation's cybersecurity. FireEye, which provides detection and response to advanced cyber threats, data breaches, and zero-day attacks, uses various sources such as security consultation, a global network of sensors, and worldwide intelligence analysts to constantly evaluate and adjust the view of the threat landscape [11, 24]. The DoD must also follow suit by employing constant evaluation and adjustment of the cyber threat landscape in the face of dynamic cyber threat tactics, which ultimately embodies the mindset of resilience engineering [6]. Cyber threats span the globe and are not affected by geographical boundaries, which often lead to a highly complex cyber attack SoS that can be difficult to assess and defend against, requiring a comprehensive and multifaceted solution [11]. For this reason, partnerships must be established between the Department of Defense and the private sector – and possibly with allied nations across the globe – in order to begin developing a fully resilient system or SoS that can compete with the rapid evolution of the cyber criminal enterprise. By merging cyber threat data from multiple organizations and nations, the likelihood of making more timely and accurate correlations between various threat activities will increase. This could be accomplished through a central repository of active cyber threat operations, such as those identified by systems like FireEye, which can automatically analyze and correlate data, followed by timely dissemination to relevant stakeholders [11]. Access to the wealth of information contained within this central repository by "cyber warriors" across the world will not only paint a broader picture of the threat landscape but also aid in more rapid development of future cyber-secure systems that can be reconfigured and restored with minimal impact following an unanticipated disturbance. If cyberthreat knowledge continues to be closely held among organizations and allied nations, the cyber threat will continue to exploit system vulnerabilities for those organizations and nations that simply do not have the resources (e.g., knowledge and finances) to develop or maintain a resilient system in the face of emerging cyber threats. This bridge to the private sector and allied nations will surely require sensitivity with respect to the protection of privacy and civil liberties. Currently, these factors are accounted for under the Fair Information Practice Principles, but it will be imperative that as partnerships grow in scale and depth, these frameworks must be reevaluated to ensure continued applicability [2].

11.6 Needed Shift from Survivable to Resilient Cyber-Secure DoD Systems

Just as the US military modifies its tactics, techniques, and procedures (TTPs) to a changing threat environment, so do our adversaries in an effort to continue evading and overcoming our cyber defensive measures – this is particularly the case as the "threat tempo" increases, resulting from the rapid development of technology [25]. The concept of operations (CONOPS) by which we combat kinetic attacks through Joint Operations can be leveraged for combating cyber attacks. These Joint Operations are defined by two or more departments acting in concert with each other to carry out a military action and serve as the primary method for combat operations in support of Department of Defense objectives [26, 27]. Survivability is critical during military operations as it correlates to the ability to withstand an attack; however, it does not include the ability to bounce back after an attack [6]. This is where the shift to resilience – or more specifically, the ability to learn from and adapt to a disruption – becomes crucial for cyber warfare [6]. As can be expected, as complexity increases in the cyber environment and in the systems used to protect against and recover from cyber threats in an elastic manner, trade-offs will no doubt be necessary just as they are during wartime operations. For instance, in the field of cybersecurity, where antivirus scripts are ubiquitous, a trade-off between the level of protection against malicious software and the affordability of such antivirus software as well as lack of interruption to operations for the customer must be made. Through collaboration of multiple industries, similar to a joint military operation, antivirus software can be greatly improved by using best practices and timely and relevant transference of threat data to more rapidly return a system to its operational state following a disruption rather than simply trying to prevent a disruption from occurring.

11.7 Conclusion

Efforts are currently being pursued to enhance the ability to prevent and minimize damage resulting from a cyber attack on DoD systems and SoS. However, as previously discussed, measures solely intended to prevent disruptions will not suffice in the long-term battle against cyber threats as they will continue to evolve at a pace greater than can be combated. For that reason, a resilient DoD cyber-secure SoS that embraces cyber attacks and the inevitability of system failure is imperative. These systems and system of systems must be founded in a framework of collaborative information sharing between the private industry and the various government agencies charged to protect critical infrastructure against cyber threats. Furthermore, systems must be in place that improve awareness and recognition of cyber threats among system users thorough compelling and nontedious training, as well as systems such as enhanced antivirus software that are capable of reducing the success of cyber attacks in addition to aiding in system recovery to normal operations. Lastly, a system is needed that can train and retain DoD "cyber warriors" with the skills necessary to increase the speed and probability that systems will recover following a cyber attack. These systems must be developed with the concept of resilience remaining prevalent throughout the entire system life cycle; especially as trade-offs between resilience and affordability become inevitable, which often have a direct impact on system sustainment [22].

References

1. Exec. Order No. 13010: Critical Infrastructure Protection, 3 C.F.R. (1996)
2. Exec. Order No. 13636: Improving Critical Infrastructure Cybersecurity, 3 C.F.R. (2013)
3. Emerging Cyber Threats, Homeland Security Cong., 7 (2016) (testimony of Michael McCaul, R-TX)
4. Carter A (2015) United States of America. The Department of Defense. The Department of Defense Cyber Strategy. Washington DC, April 2015
5. Madni AM (2017) Transdisciplinary systems engineering: exploiting convergence in a hyperconnected world, Springer, 2017 January Release
6. Madni AM, Jackson S (2009) Towards a conceptual framework for resilience engineering. IEEE Syst J 3(2):181–191
7. Madni AM (2016) Course review. Engineered resilient systems and system-of-systems. University of Southern California, Olin Hall, Los Angeles, 25 Apr 2016, Lecture
8. Leonard WM (2016) SAE 599 midterm. University of Southern California, 11 Apr 2016
9. Elkus A (2015) The devastating breach of US government data highlights an illusory cybersecurity paradox. Business Insider. Business Insider, Inc, 18 June 2015
10. United States of America. Department of Defense. Office of The Secretary of Defense. Department of Defense Cybersecurity Culture and Compliance Initiative (DC3I). By Michael L. Bruhn, General Martin E. Dempsey, and Ash Carter. Washington, DC, 30 Sept 2015
11. Emerging Cyber Threats, Homeland Security Cong., 7 (2016) (testimony of Jennifer Kolde, Technical Director, FireEye, Inc.)
12. About the Department of Defense (DoD). U.S. Department of Defense, 27 Aug 2015. Web. http://www.defense.gov/About-DoD
13. Special Report: Cyber Strategy. U.S. Department of Defense, Apr 2015. Web http://www.defense.gov/News/Special-Reports/0415_Cyber-Strategy
14. Senate Testimony: Cyber Security as a Strategic Concern. Senate testimony: cyber security as a strategic concern. Headline News, 2 Feb 2012. Web. http://www.infosecisland.com/blogview/19888-Senate-Testimony-Cyber-Security-as-a-Strategic-Concern.html
15. Office of the Director of National Intelligence. Unclassified statement for the record on the worldwide threat assessment of the US Intelligence Community for the Senate Select Committee on Intelligence, at 8 (Jan 31, 2012)

16. Department of Defense. Chief Information Officer. DoD strategy for defending networks, systems, and data. 13 Nov 2013
17. The DoD Cybersecurity Policy Chart. CSIAC – information assurance. 27 Oct 2015. http://iac.dtic.mil/csiac/ia_policychart.html
18. Madni AM (2016) Resilience concepts and current limitation. Engineered resilient systems and system-of-systems. University of Southern California, Olin Hall, Los Angeles, 25 Jan 2016, Lecture
19. Rsignia Inc. Rsignia Inc. and Air Force Research Laboratory Enter Agreement to Research Cyber Security Capabilities. PRNewswire, 7 June 2016. http://www.prnewswire.com/news-releases/rsignia-inc-and-air-force-research-laboratory-enter-agreement-to-research-cyber-security-capabilities-582135071.html
20. Reports Reveal Four Cyber Security Trends–and the Need for Better Cyber Security Training & Awareness. Cybersecurity Bulletin (May 2016)
21. Rawlins LK (2016) The 'Big Five' national cyber security projects. ITWeb Security. 18 May 2016. http://www.itweb.co.za/index.php?option=com_content&view=article&id=152608
22. Wheaton MJ (2016) Affordable resilient systems. Engineered resilient systems and system-of-systems. University of Southern California, Olin Hall, Los Angeles, 4 Apr 2016, Lecture
23. Madni AM (2016) Systems thinking. Engineered resilient systems and system-of-systems. University of Southern California, Olin Hall, Los Angeles, 8 Feb 2016. Lecture
24. Why fireeye: the premier cyber security company that protects you before, during and after a breach. FireEye. Web. https://www.fireeye.com/company/why-fireeye.html
25. Emerging Cyber Threats, Homeland Security Cong., 7 (2016) (testimony of Michael McCaul, R-TX)
26. Leonard WM (2016) Introducing resilience into the command and control of joint operations. University of Southern California, 17 May 2016, SAE 599 Final Term Paper
27. Publication, Joint. 3–0 (1995) Doctrine for joint operations. Joint Chiefs of Staff, Washington, DC
28. Madni AM (2012) Adaptable platform-based engineering: key enablers and outlook for the future. Syst Eng 15(1):95–107
29. Clapper JR (2016) United States of America. Office of the Director of National Intelligence. Statement for the Record: Worldwide Threat Assessment of the US Intelligence Community, 9 Feb 2016

Chapter 12
Architecting Cyber-Secure, Resilient System-of-Systems

Kurt Klingensmith and Azad M. Madni

Abstract The DoD system-of-systems (SoS) relies heavily on cyberspace operations. The latter tend to be vulnerable to a variety of disruptions. These disruptions can be from within or outside the SoS. The ability to withstand disruptions is essential to maintaining a competitive edge in terms of freedom of maneuver afforded by cyberspace. Since impenetrable cyberspace capacity is implausible, architecting for cyber-resilience has become a national imperative. This paper explores the complexity of cybersecurity and ways to achieve cyber-resilience that is informed by cyber strategies and techniques developed within a model-based engineering framework.

Keywords Cybersecurity • Resilience • System-of-systems • Architecting

12.1 Introduction

Technological maturation continues to transform military operations and capabilities. In the past two decades, the concept of Net-Centric Warfare (NCW) emerged [1] to "harness the power of information and network connectivity" [2]. NCW technology allowed normally disparate systems to share a rapidly formed, current common informational picture. However, harnessing this capability relies on an overarching domain with fuzzy boundaries. This domain is cyberspace, "one of [the] five interdependent domains" of military operations [3]. The DoD's Joint Publication 3-12R defines cyberspace as:

"…the global domain within the information environment consisting of the interdependent network of information technology infrastructures and resident data, including the Internet, telecommunications networks, computer systems, and embedded processors and controllers." – Joint Publication 3-12R [3]

K. Klingensmith
US Army, Monterey, CA, USA
e-mail: kurt.m.klingensmith.mil@mail.mil

A.M. Madni (✉)
University of Southern California, Los Angeles, CA, USA
e-mail: azad.madni@usc.edu

A.M. Madni et al. (eds.), *Disciplinary Convergence in Systems Engineering Research*, https://doi.org/10.1007/978-3-319-62217-0_12

Unique to cyberspace, though, is the fact that technological maturation touches more than just military institutions and systems. Increasing capabilities and falling device prices have opened the domain for different types of conflicts ranging from conventional, military agents to asymmetric, informal entities [4]. Cyberspace is a socio-technical system of systems (ST-SoS) that comprises systems (including automated software systems) and human agents [5]. Given the complexity of socio-technical systems (STS), it becomes difficult to understand, evaluate, and appropriately leverage one's capacity to effectively operate offensively and defensively in cyberspace. Singer and Friedman suggest that humans are the primary source of complexity, and when considering cybersecurity, "[t]he people behind the machines are inherently inside any problem or needed solution" [5].

There is no technological silver bullet for cybersecurity. The evolving nature of vulnerabilities, vectors, and exploits across complex ST-SoS quickly erodes any technological advantage [6]. This recognition is not intended to discourage software and hardware advancement; rather, it is a reminder to cyber planners to employ a holistic security approach that acknowledges the inevitability of both expected and unexpected external and systemic disruptions [7]. This paper explores the complexity aspects of cybersecurity within a model-based framework for architecting cyber-resilient systems. The latter is guided by the DoD Cyber Strategy and informed by applicable research on resilience and cyberspace operations [8].

12.2 The Need for Resilience

12.2.1 Complexity and Interdependence

The state of cybersecurity for a military unit depends on various factors, many of which influence each other on the path toward strengthening (or eroding) cybersecurity. Figure 12.1 depicts the interdependent influencing agents and how they contribute to cybersecurity. The figure delineates between the DoD-controlled enterprise in green, with the gray areas representing the extended enterprise, adjacent systems, and external entities influencing DoD cybersecurity. Elements within the DoD's enterprise spill over into the extended enterprise, such as outsourced or contracted personnel, or private industry developing components for DoD systems. The cyber adversary has potential reach across the spectrum; for example, a vulnerability in a private manufacturing process of a new military system may lead to successful cyber espionage by the adversary. This degrades cyber capacity through loss of an emerging system capability, while also reshaping the scope of possible missions due to system limitations or delays as the system goes through added development to regain its competitive edge. Fixing this may incur changes in budget policies and laws, resulting in lost opportunity cost as other cybersecurity-enhancing alternatives go unfunded.

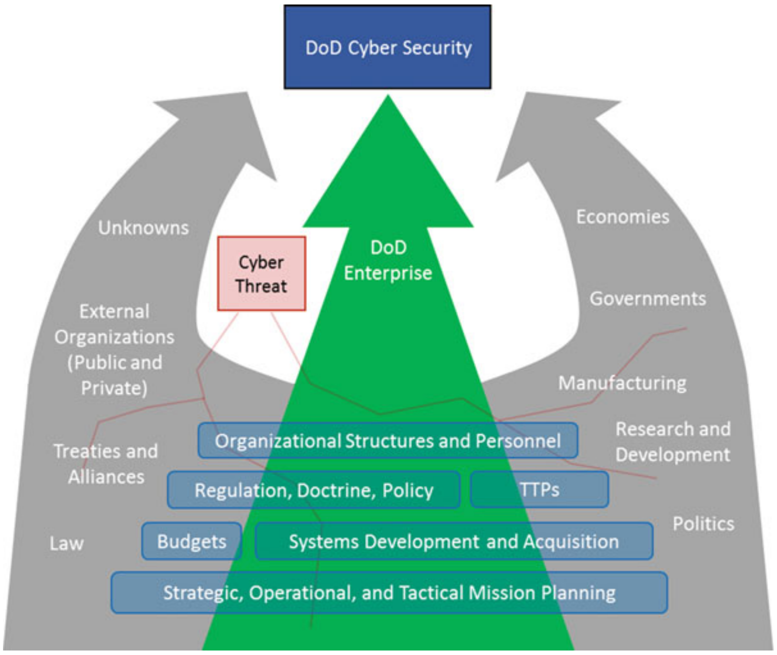

DoD Cyber Security
Unknowns
Cyber Threat
DoD Enterprise
Economies
Governments
External Organizations (Public and Private)
Manufacturing
Research and Development
Treaties and Alliances
Organizational Structures and Personnel
Regulation, Doctrine, Policy
TTPs
Law
Budgets
Systems Development and Acquisition
Politics
Strategic, Operational, and Tactical Mission Planning

Fig. 12.1 DoD cybersecurity and its influencers

It is important to note that what is not shown in Fig. 12.1 is that a related web exists for the cyber adversary's capacity. Also, each element influences each other; economies impact governments, governments impact treaties and alliances, and law impacts external organizations. This change cascade spreads to elements under the DoD's control and influence. For example, a treaty may constrain mission planning, or new law might dictate organizational structures. Furthermore, each element has an extended chain of items that feed into them. Research and development may create tools and techniques for offensive and defensive cyber operations, but research and development presents a cyber espionage opportunity. Such espionage could discover Zero Day exploits in new systems, which are "previously unknown vulnerabilities" [5]. In other words, research and development could inadvertently deliver a supposed cutting-edge cybersecurity product for which the cyber adversary already knows of an exploitable vulnerability.

Similar problems may exist in manufacturing and in external agencies such as private firms and industry. Here, a hardware supply chain for a manufacturer may extend into less regulated markets, resulting in "counterfeit parts… in the supply chain of national security-related electronic equipment" [9]. In 2008, non-stealth Israeli fighter jets infiltrated Syrian airspace undetected, giving rise to speculation that "commercial off-the-shelf [COTS] microprocessors in the Syrian radar [had] been purposely fabricated with a hidden 'backdoor' inside" [10]. If true, a component sourced via a nonmilitary supplier created a vulnerability in the air defense

SoS's cyber-attack surface, the chip became a vector via offensive cyber operations, and the vulnerability served as a basis for an exploit upon deception or deactivation of the air defense SoS [6]. Offsetting such a risk, though, incurs cascading trade-offs. Testing all components in COTS DoD items, if even possible, will incur great cost and time. Manufacturing all components within one nation depends upon that nation's resources and knowledge capacity to compete with foreign COTS alternatives. Even then, cyber compromise is still a possibility; the F-35 fighter jet under development in the US, still experienced "a spate of attacks targeting... [its] design and manufacturing process" [5].

The preceding highlights examples of emergence within the complex ST-SoS of cyberspace, where interplay amongst the heterogeneous constituent subsystems and human agents gives rise to otherwise impossible functions and capabilities [11]. Predicting if and how emergence will occur, though, is challenging for an ST-SoS such as cyberspace. Furthermore, constituent system coupling ranges from loose to strict and even strictly coupled systems can devolve, as malicious actors exploit cyber vulnerabilities within a system. Despite the negative emergent possibilities, the benefits of NCW, networked cyber-physical systems (CPS), and broad connectivity represent positive emergence that provides advantages for the DoD. Gaining this competitive edge necessitates a cyberspace enclave large enough to provide the emergence necessary to achieve the SoS objectives. Singer and Friedman assert that a network's "security is generally inversely correlated with size, while network utility is positively correlated" [5]. However, a more appropriate assertion would be to replace "network security" with "ease of securing," as it is no given that large networks are automatically insecure. Adapting Singer and Friedman's concept and mapping it to emergent behavior yields Fig. 12.2, which further highlights the complexity of cybersecurity brought forth by Fig. 12.1 [5]. In fact, at a certain level of abstraction, cybersecurity becomes a wicked problem due to its irreconcilable interdependencies and counterintuitive state changes and propagations [12].

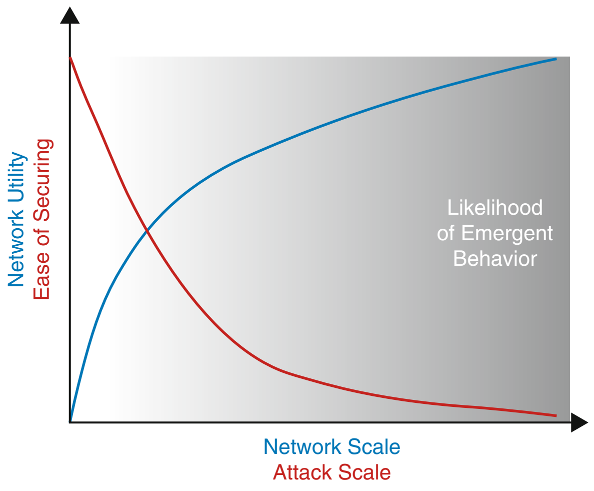


Fig. 12.2 The relationship between network scale/attack surface, network utility/ease of securing, and emergence

12.2.2 Defining Resilience for Cyber-Secure Systems

Given the volatile complexity of the ST-SoS, cyber disruption is highly probable, especially for DoD operations dependent upon net-enabled capabilities. One of the challenging aspects of cybersecurity is that the types of disruptions, external and systemic, have blurred boundaries and definitions. While traditional external disruptions impact cybersecurity (world events, natural disasters, status of enabling systems such as power distribution, etc.), external elements exploit systemic vulnerabilities, resulting in external elements entering the SoS of interest, both becoming systemic disruptions and creating new, unique systemic disruptions [7].

For example, consider an Advanced Persistent Threat (APT). The APT is an organized element targeting a specific entity in cyberspace over a protracted period, embodying the "professionalization of cyberattacks" [13]. An APT exploits systemic vulnerabilities, and may use "spear phishing" to obtain network access credentials from a specific, unwitting human agent within the SoS [5]. Or, an APT may leverage an external disruption such as a recession to exploit a financially struggling worker. Another such example may be manipulating an enabling system such as a COTS antivirus provider's virus definition files so as to create a backdoor [14]. Regardless, the external element finds or creates a systemic vulnerability, turning it into a vector in which to enter the cyberspace enclave of the ST-SoS. At this point, it may launch any number of exploits, be it confidentiality-disrupting data exfiltration or the launch of an automated software agent within the system. This problem set resembles the "Swiss cheese model of system accidents," though with some domain-specific variations [15]. "Active failures" correlate to human agent disruptions, while "Latent conditions" represent system vulnerabilities that could provoke or enable disruption [15]. Cyber adversaries may manipulate both or create and inject new latent conditions expressly for exploitation. Additionally, deceptive techniques could manipulate systems to control and align various holes across the enterprise to achieve a desired, cross-organizational reach that goes undetected.

Given the inevitability of varied, adaptive, and deceptive disruptions, resilience becomes a necessity. Goerger, Madni, and Eslinger adapt Neches' definition of resilience to yield the following [16, 17]:

> *"A resilient system is trusted and effective out of the box, can be used in a wide range of contexts, is easily adapted to many others through reconfiguration and/or replacement, and has a graceful and detectable degradation of function."* – Definition of resilience [17]

There are several resilience characteristics that enable the preceding definition. Fusing the many faces of resilience [7] with DoD resilient system properties [17] results in the following:

- *Avoidance and Anticipation* [7]: The cyber-resilient SoS maintains awareness of its environment and likely disruption sources, and flexes and adapts itself in light of anticipated disruptions.

- *Repelling, Resisting, Deterring* [17]: Repelling and resisting enable a cyber-resilient SoS to preserve its cyberspace boundaries and interfaces [17], while a truly cyber-resilient SoS will deter adversarial disruptions and thus reinforce the ability to repel and resist.
- *Withstanding, Absorbing* [7]: A cyber-resilient SoS will render intrusion ineffective, making the SoS an asymptomatic carrier of a foreign intruder or foreign software and hardware.
- *Adaptation* [7, 17]: The cyber-resilient SoS makes dynamic adjustments to its architecture, constituent systems, and behavior so as to maintain sufficient functionality and capability despite disruptions [7, 17]. Adaptability requires self- and contextual-awareness and learning capacity.
- *Recovering* [7, 17]: The SoS restores itself based upon its learning from before, during, and after the disruption [7]. Recovery leverages adaptability and flexibility to reconstitute and perform at a level higher than its predisruption state [18]. Resilient learning systems exploit failed disruptions, improving their cybersecurity posture as a result of each survived disruption.
- *Flexibility and Adaptability* [18]: Both attributes allow architectural reconfigurability, with flexible architectures accommodating expected changes (software versions, hardware increments), while adaptable architectures accommodate unexpected change [18].

Figure 12.3 depicts how these attributes and characteristics create a cyber-resilient system over the course of a disruption. The learning, cyber-resilient SoS functions well above the minimum required threshold for its current use case. Upon

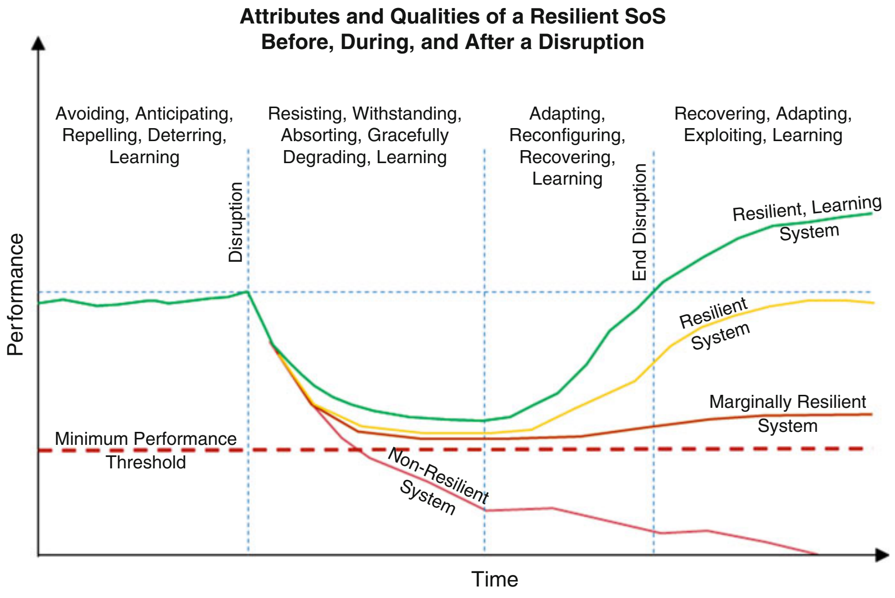


Fig. 12.3 The cyber-resilient SoS's performance versus a nonresilient system

encountering a disruption, after initial rapid degradation, the SoS begins to control its degradation via absorption and other techniques, until it eventually reaches and surpasses predisruption performance. The nonresilient system is unable to correct its decaying performance, eventually failing. As an example, consider a joint coalition dependent upon GPS for coordinating movements and targeting; a cyber-attack modifies stored GPS grids, which may deceive the coalition. The resilient coalition has systems that absorb, control degradation, and adapt to validate and correct the data, whilst the nonresilient system ultimately fails. The cyber-resilient system's minimum performance threshold is set by the mission planners, who must determine what cyberspace functions are necessary for a given use case and mission. Key to determining this threshold is understanding the context so as to architect and design the SoS to "survive most likely and worst case scenarios, either natural or man-made" [7].

12.3 Architecting for Cyber-Resilience

System advances continue to address cybersecurity through means such as building and refining software, hardware, organizational and operational processes, and private, public, and government agencies. However, simply choosing various optimized COTS solutions does not guarantee a cyber-resilient Cyber-Physical SoS (CP-SoS). In some cases, optimization in one area may create issues in other areas, inadvertently moving the system toward a state of "brittleness," a nonresilient state susceptible to failure from cyber disruptions [7]. Thus, dependent upon the objectives and the abstraction level, suboptimal elements and constituent systems that integrate synergistically may yield greater resilience. Key to this is understanding the boundary for the SoS and then making appropriate trade-offs that holistically optimize cybersecurity in light of the SoS objectives. What follows are resilience techniques to attain this.

12.3.1 Cyber Self-Awareness

Adding resilience to the SoS requires self-awareness of the SoS and its potential use cases and contexts, as well as the associated threats and vulnerabilities. Jabbour and Poisson state that conducting a Cyber Vulnerability Assessment (CVA) is "an exercise in the knowledge of us" [19]. This may start with mapping the SoS's residence along RAND's interoperability levels (strategic, operational, and tactical) [20]. Strategic-level vulnerabilities and concerns vary from operational and tactical concerns, while technology binds the three levels together [20]. Each level and the systems at play within cyberspace vary; awareness of friendly disposition within cyberspace is critical to understanding enemy and threat disposition, as the scope and type of likely threats and attacks shift. A complicating characteristic of cyberspace is its global interconnectedness. A system malware intrusion at the tactical level may have little to no impact on immediate operations, but the malware may spread through global links and attempt to create backdoors across large-scale, strategic infrastructure. An example of this is believed to have occurred when an "unwitting soldier [forward] in Afghanistan" inserted "an infected USB drive" that set off malware replication with global reach in the DoD GIG, resulting in strategic refocusing and policy change [21]. The inverse was already discussed with the Israeli operation, in which the strategic enterprise sourced vulnerable components that were then exploited for tactical advantages [10].

Thus, the "exercise in the knowledge of us" can be a counterintuitive game [19]. This is why CVAs are critical to selecting resilience techniques. CVAs case the attack surface of a CP-SoS and assess the cyber topology. These assessments may be done by humans as well as intelligent systems, and must be conducted continuously. An effective CVA nests itself with higher and lower echelons so as to understand relevant vulnerabilities that may flow into the CP-SoS's boundary. Jabbour and Muccio state that "reliance of a Mission Essential Function (MEF) on cyberspace makes cyberspace a center of gravity an adversary may exploit," thus "directly [engaging] the MEF without employment of . . .forces or weapons" [22]. Accordingly, mapping of MEFs for a CP-SoS and its mission are critical, especially in terms of understanding and envisioning the layered nature of vulnerabilities and threats across the operational enterprise. This relates to the minimum performance threshold in Fig. 12.3; understanding how to define that threshold and how the SoS relies on it is key to resilience and mission success. Vulnerabilities and applicable threats depend upon what one is trying to do and with what equipment and methods. Such decisions can limit the scope of possible system states in such a way that an SoS may more readily drift toward brittleness and failure with less effort from cyber-attackers.

12.3.2 Resilience Techniques and Principles for Cyber-Secure Systems-of-Systems

Baseline Network and CP-SoS Composition. The network technologies and systems selected set the baseline for cybersecurity. And once systems are chosen, proper configuration must follow. An Australian DoD study found that 85% of its cyber vulnerabilities were reduced through four strategies: "application whitelisting; patching applications and operating systems[;] using the latest version; and minimizing administrative privileges" [23, 24]. These basic concepts cleared the noise of low-sophistication, low-risk actors from the threat spectrum while preventing rapid and easy exploitation from threats such as APTs. They can also block known "threat signatures" to prevent easy reuse of techniques and methods [21]. Such measures, though, can be difficult in a heterogeneous CP-SoS. Baseline methods may ensure good password protection against prohibited uses, but did the maintenance team reset default passwords and patch the mechatronic control units layered across a CP-SoS? A desktop computer may be inaccessible, but a router still set with a default password or a CPS control unit locked with password "123456" represents an easily avoidable, hidden vulnerability [5]. Closing these gaps avoids, repels, and deters the opportunists and denies APTs an easy day, buying time that contributes to graceful degradation in the face of disruption.

Measured Heterogeneity Classical methods such as fault-tolerance and redundancy "assume some natural disaster, accident failure, or crises rather than deliberate attack" [5]. And while they contribute to resilience in general, redundancy can work for and against network resilience. Having homogenous backups and alternate paths may work well for nonmalicious or static disruptions, but an intelligent adversary gains an advantage from this setup: potential uniformity of attack surface, and thus uniformity of vulnerabilities across the SoS. Homogeneity makes patching, versioning, and life cycle management easier, thus contributing to baseline threat mitigation [23, 24]. However, a vulnerability, especially a Zero Day vulnerability, will now be present across the entire CP-SoS, turning redundant paths and systems into extra vectors. Conversely, increased heterogeneity provides adversaries with more vulnerabilities to target, though the reach provided by exploiting a vulnerability may be lower. Furthermore, heterogeneity adds complication to the enterprise and its managers. Thus, system architects must make the trade-off appropriate to deliver a measured application of heterogeneity across networks and constituent systems for a given SoS context. Appropriate heterogeneity of functionality and system type (to include redundant functionality and systems) will avoid rapid degradation, drift toward brittleness, and failure during disruption.

Architectural Configuration and Adaptability. The SoS Architecture, consisting of its functions and interfaces, decomposes into web constituent systems, subsystems, and components. Preemptive architectural techniques, such as "Segmentation, Isolation, [and] Containment" serve to map functions and interfaces in a manner that compartmentalizes "components of dubious pedigree from those trusted, to reduce the attack surface and limit the damage of exploits when they occur" [25]. This is a passive and dynamic strategy, with the baseline architecture being a planned element and system adaptability and flexibility dynamically allowing for reconfiguration during use [25]. Passively, interfaces may be limited to less defensible legacy constituents with internal partitioning for information flow gatekeeping. Dynamically, sensitive data storage may shift given system needs or in response to detection of an active intrusion. Or, distributed computing techniques may cross-level software functionality to compensate for a compromised or lost CPS/node, shifting interfaces, and functions mid-mission.
Related to this is control of interfaces and Information Exchange Boundaries (IEB) [19]. Architectural interfaces in a cyber-physical and ST-SoS include human-system, hardware, and software boundaries. An IEB is a type of architectural interface in which information relevant to system operation, physical and cyber,

transfers between elements within the architecture [19]. This could be data packets across Ethernet cables or physical hardware interaction that is converted to data. Also key to understanding these interfaces is the understanding of interface persistence and "nonpersistence" [25]. Not all interfaces need to remain active, reducing system data infiltration and exfiltration points as well as vectors for influencing the system [25].

Demonstration of interface management and nonpersistence's impact on cybersecurity came when, on a traveling Jeep, demonstrators remotely disabled the brakes, the transmission, and turned on the windshield wipers all through an Internet connection in the infotainment system [26, 27]. Nonpersistence may have denied the external Internet link during nonessential times, while interface management may have internally partitioned cyber-physical and mechatronic control systems so that a compromised stereo system would not have a clear path to a functionally critical braking system. Front-end systems architecting would address whether or not an interface between brakes and an entertainment system is necessary in such a CPS. Consider that these are known interfaces; system self-awareness will allow a knowledge of current interfaces and their status (e.g., an unused Bluetooth receiver that is on and searchable by local devices). Furthermore, a self-aware system would sense newly adapted interfaces, paying homage to Eberhardt Rechtin's heuristic: "Be prepared for reality to add a few interfaces of its own" [28].

Automation. A Zero Day vulnerability may go undiscovered until an adversary exploits it, while self-replicating malware may spread across many systems at a rate greater than what humans could attain through manual installation. Often, the attack is found after the adversary successfully completes "Actions on Objectives," the final step in the Lockheed Martin Cyber Kill Chain [29]. Automated systems provide a logical counter to this, especially when coupled with machine learning. In this sense, the role of automation is to self-regulate cyberspace occupied by the SoS and maintain an awareness of all activities and potential activities within the SoS. This means automated CVAs and interface assessments, to include interface persistence management. It also means conducting automated "Red Team" assessments of the SoS [5, 6, 19, 21]. Such assessments can be conducted by the SoS locally or remotely via network links.

In such scenarios, virtual Red Teams could leverage automatically generated attack trees to case the SoS's cyberspace footprint for pathways that an adversary may use to successfully maneuver through the network and the Cyber Kill Chain [29, 30]. There are limits to this; virtually casing scenarios involving human agent vulnerabilities will be unfeasible for most real-time, dynamic SoS operations. Such a scenario would be more likely for pre-SoS deployment, in which case, traditional Red Teams could fulfill the requirement. Virtual, automated CVAs and Red Team cyber-gaming may have an easier time discerning technical, systemic vulnerabilities, such as misconfigured systems with improper patches or open ports. The Defense Information Systems Agency (DISA) offers a similar solution known as the "Assured Compliance Assessment Solution" (ACAS), which scans networks for vulnerabilities and configuration issues while providing a security assessment to its users [31].

Biomimicry. The Artificial Immune System (AIS) concept virtualizes the biological immune system's (BIS) "multilayer protection system" [32]. In addition to baseline defenses to ward off threats, or pathogens, the AIS must "recognize all cells… within the body and categorize those cells as self or non-self," taking appropriate action based upon the classification [32]. This allows for detection of intruders while maintaining awareness of normal network operations. Nonself elements are removed, modified (neutralized), or quarantined, rendering their presence within the system asymptomatic. Spam email filtering provides an example of this, in which heuristic guidance libraries complement a "Bayesian token library" for probabilistic analysis and "English word libraries" for content analysis [32]. When combined, the libraries effectively screen malicious spam email from normal traffic, thus eliminating the effects of the attack [32].

In a dynamic SoS, though, normal data traffic, data patterns, and SoS members will routinely shift during the course of operations. Furthermore, clever adversaries may become aware of the presence of an AIS and adapt their techniques so as to mask their behaviors similar to expected behaviors of the SoS's constituent systems and data flows [5]. So while countering systems such as antivirus may "use 'heuristic' detections to identify suspicious computer code behavior," those heuristics and rules delineating between self and adversary must evolve [5]. There is also the threat of the AIS itself being compromised, in which the attack may deceive the SoS by making the AIS confuse self and nonself entities. Thus, the AIS must learn about itself and adapt in ways similar to how the SoS and the threats to the SoS adapt.

Data and Knowledge Management. Rapid sharing of relevant data amongst a CP-SoS creates advantageous emergence, but those data and their management may contribute to the cyber-attack surface. Data are also a vehicle for undesirable SoS emergence. Data come in many forms, some of it valuable information for intruders to target and some of it valuable for modifying so as to create undesirable emergence. Complicating this is the zero latency enterprise characteristic of a CP-SoS, in that "all parts of the enterprise can respond to events as soon as they become known to any part of the enterprise" [33]. This was the emergent benefit of net-centricity, but the advantage depends upon data confidentiality, integrity, and availability [19]. Actively partitioning global data across local caches allows persistent access amidst link disruptions [33]. Probabilistic analysis and machine reasoning helps constituent systems determine information relevance when deciding what to pull forward to local caches; conversely, it also helps them determine how to route generated and collected information [33].

Certain elements of the SoS may face greater likelihood of an attack on their data stores. A resilient SoS would intelligently shift where and how critical data are stored so as to decrease the vulnerability and risk to the SoS due to a few highly vulnerable nodes. Data would actively shift between remote repositories, clouds, and local stores and caches as necessary to achieve SoS objectives in light of potential disruptions to data confidentiality, integrity, and availability. Note that network resilience plays a role, as data shifting depends upon link persistence amongst nodes with mission-relevant data. A cyber-attack that disrupts a vital communication link could be mitigated via data resilience, in which the network repositioned critical data based on a self-CVA so as to nullify brittleness from relying on too few data paths.

Actual data storage needs to use resilient methods. Shifting data amongst constituents in the cloud creates excess copies that may add attack surface of the SoS. Data wiping as material shifts needs to be sufficient so as to leave no possibility of data exfiltration. Data tracking must also exist at the IEBs, which can couple with the AIS to analyze typical self-data movement versus nonself. This is critical, as APT detection mostly occurs at the "exfiltration phase, when massive amounts of data leave the network" [5]. Data manipulation from an adversary can compromise data integrity, leading to deception and poor SoS functionality and decision-making. Thus, in a mobile SoS mesh, cloud storage must emulate concepts from RAID (Redundant Array of Independent Disks) [34], "which creates reliable storage arrays from unreliable hard drives" [35]. In this case, those independent disks or unreliable hard drives are constituent systems in a distributed CP-SoS, each of which can locally store partitioned data while also reaching back to large data repositories [34, 35]. Data and knowledge management resilience represents a trade-off, though, as data replication and movement increases attack surface for data while adding complexity. Planners must decide which data may only reside in specific locales and which data may flow freely in the SoS.

Active Defense. A resilient system repels and deters disruptions. Baseline defensive and protective methods ensure a robust, resilient network that may ward off the majority of attackers, but there remains a threat from focused adversaries, APTs, and exploitable Zero Day vulnerabilities. In a military environment, persistent and focused adversaries will be unavoidable. A counter to this is the concept of "a hostile work environment" for intruders [21]. This is analogous to the land warfare technique of obstacles and minefields. Tracts of terrain may not be actively defended territory, yet they are extremely unforgiving to outsiders that do not know where mines lie. Meanwhile, obstacles limit movement or, even worse, direct movement to minefields or terrain that is tactically undesirable. This has the effect of deincentivizing movement in the area, with any enemy movement subject to exploitation. In cyberspace, creation of "a hostile environment [is done through implanting] malware in honeypots" [21]. In addition to malware honeypots, misleading or intentionally false information could exist so as to delay or disrupt the intruder's actions. These concepts create time for the defender, allowing the SoS to identify the threat, delay and inconsequentially absorb the threat, and then adapt so as to mitigate or eliminate the effects of the threat. Furthermore, identifying the threat during this time could allow active targeting and retaliation [21].

Human Integration. Singer and Friedman assert that "[resilience] cannot be separated from the human component," mainly due to the adaptability humans bring to an SoS as well as their potential to introduce disruptions [5]. A challenge with this comes from the automated dimension of cyberspace, in which attacks and counterattacks can rapidly occur. Even if humans are aware of the activity, the rate at which humans adapt "sets an upper bound on how fast systems can adapt" [36]. In other words, human behavior and decision-making within the SoS will only synchronize with relevant automated cyber-activity if the automation adapts the SoS slow enough for humans to follow the changes [36]. However, cyber-attacks can develop at a rate beyond human perception, meaning some degree of automated response is necessary. A resilient system will automate quick responses beyond human perception to counter specific disruptions; in other situations, it will absorb and resist to gain sufficient time so as to bring the human into the decision cycle. Systems architects and cyber planners must understand and then classify which scenarios warrant which responses; continuous CVAs will feed into this evolving classification.

Human integration also ties into the concept of Cyber Situational Awareness (CSA) [37]. Matthews, Arata, and Hale explain that CSA concerns "perception of the [cyber] surroundings and derivative implications critical to decision makers in complex, dynamic areas" [37]. They further assert that CSA hinges upon the following factors: "intelligence, integration, speed, analytics, expertise, and resiliency" [37]. While listed as the final component, resilience is a cumulative quality attribute drawing from the other five factors listed. Humans also play a critical role in these factors, ultimately tying them together to contribute to cyber-resilience.

Humans also factor in how they are literally integrated into the system, to include authentication, task allocation, data access/write privileges, and responsibilities within the network. This integration and access impacts confidentiality, integrity, and availability of network data and services, both inadvertently and advertently through challenges such as the insider threat [38]. Partitioning access control to data and enclaves within the occupied cyberspace for the SoS can help absorb or delay the disruption caused by an insider threat. Conversely, overpartitioning may negate some of the rapid adaptability possessed by human agents due to system-imposed constraints. This is an area where proper management of the enterprise and extended enterprise (in this case, human resources management) increases resilience.

12.4 A Framework for Architecting Resilience

12.4.1 The Framework

The requirements for a resilience engineering framework are associated with use scenarios, exploration of resilience concepts and mechanisms, resilience mechanism efficacy assessment, and data analytics. For the latter, the data can be from a

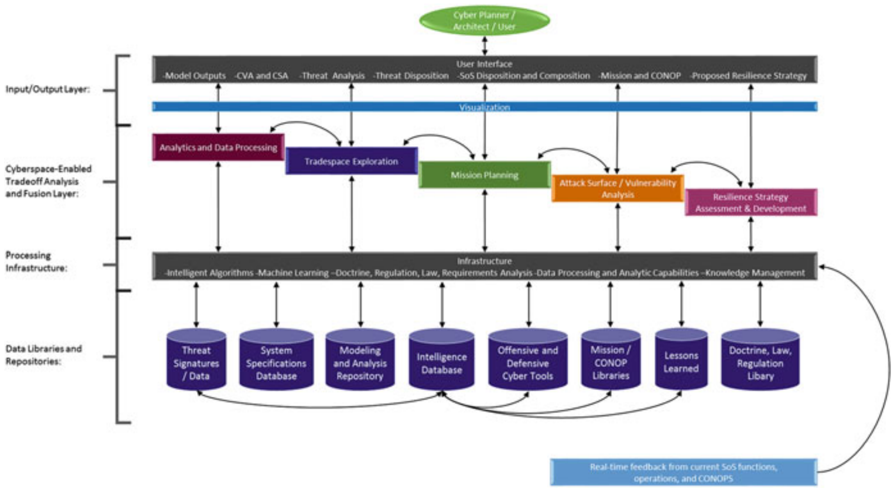

Cyber Planner / Architect / User
User Interface
-Model Outputs -CVA and CSA -Threat Analysis -Threat Disposition -SoS Disposition and Composition -Mission and CONOP -Proposed Resilience Strategy
Input/Output Layer:
Visualization
Analytics and Data Processing
Tradespace Exploration
Mission Planning
Attack Surface / Vulnerability Analysis
Resilience Strategy Assessment & Development
Cyberspace-Enabled Tradeoff Analysis and Fusion Layer:
Processing Infrastructure:
Infrastructure
-Intelligent Algorithms -Machine Learning –Doctrine, Regulation, Law, Requirements Analysis -Data Processing and Analytic Capabilities –Knowledge Management
Data Libraries and Repositories:
Threat Signatures / Data
System Specifications Database
Modeling and Analysis Repository
Intelligence Database
Offensive and Defensive Cyber Tools
Mission / CONOP Libraries
Lessons Learned
Doctrine, Law, Regulation Libary
Real-time feedback from current SoS functions, operations, and CONOPS

Fig. 12.4 Framework for architecting a cyber-resilience strategy for a complex CP-SoS

simulated or real-world operation. To support these higher-level requirements, the framework needs to offer several technical capabilities that serve as the key enablers for satisfying the high-level requirements. The technical requirements include search and retrieval, machine learning, data mining, and statistical analysis. Figure 12.4 is the result of satisfying the aforementioned requirements.

The framework has several layers, beginning with the user interaction with the Input/Output Layer. This user interface layer allows the user (who may be a cyber planner or systems architect) to see visualizations of various elements impacting cyber-resilience. The next layer down is the Cyberspace-Enabled Analysis and Fusion Layer that leverages various tools and methods to weigh information, details, and trade-offs in light of the mission and its resilience strategy. Note how each element flows into one another. The Processing Infrastructure Layer consists of the physical systems necessary to manipulate and pull relevant information for the problem. This layer reaches into the Data Libraries and Repositories, which store necessary information for analysis in the framework. Outside of the framework lies a light-blue box; this is real-time feedback from current SoS functions, operations, and CONOPS. It flows directly into the infrastructure, which aggregates data and information from live SoS operations to update data libraries and repositories as well as inform current analysis processes.

The intent of this framework is to develop a resilience strategy for a specific mission set. Not all assemblies of humans, systems, and objectives necessitate the aforementioned resilience techniques, and adding some or all may not be possible given constraints of cost, time, and subsystem capability. Thus, a mission-specific focus is necessary to achieve cyber-resilience at the appropriate level of SoS abstraction. This framework is unique to cyber planning.

12.4.2 Framework Implementation: A Use Case

Consider the previously discussed air-defense example of an Israeli cyber-attack thwarting and disabling Syrian air-defense systems [10]. Suppose a user needs to develop a cyber-resilience strategy and implementation for an air-defense system for a ground unit. Here, the user would input their mission plan at the input/output level, at which point the framework would conduct analysis in the sublayer. The Analysis layer would weigh the mission in light of data pulled from the repositories regarding threat intelligence, past missions and CONOPs, and available cyber tools (both offensive and defensive), given the constituents partaking in the SoS for the given mission. The framework would also reference active or ongoing operations for pertinent data. Then, the processing infrastructure and the analysis and fusion layer would finalize their assessments, creating visualizations that are pushed back to the user-level. The user can then accept the given products or further tweak the variables to seek different outputs from the framework. In such an example, the framework may give the following output:

- *Network Architecture:* Virtually partition more vulnerable legacy systems from current systems. Institute a complete physical gap if possible amongst critical air-threat sensing nodes to prevent a shared vulnerability from spreading instantaneously across the SoS.
- *AIS Implementation:* Run an AIS emphasizing awareness of self. Jabbour and Poisson ask if one would rather have a continuously available radar "with a random 10 percent of the displayed information inaccurate," versus having a radar "that is available 90 percent of the time with all the displayed information accurate" [20]. Deceptive feedback or limited availability is a big challenge for cyber-resilience in an SoS tasked with detecting enemy presence in a sector. The key is that the systems understand when they and their users can rely on their functionality and outputs. The AIS, tailored here to specifically analyze self-functionality, will focus on identifying deception and alerting other constituent systems and human agents of unreliable SoS data outputs. Or, it will sense movement toward nonavailability, alerting the system to adapt its architecture to maintain functionality via alternate means. The key is not just seeking redundancy and perfect availability/reliability, but ensuring that the SoS meets its objectives despite disruptions.
- *Baseline Readiness Assessment:* The framework will assess all elements within the SoS for the system to ensure compliance with latest patches, hardware/firmware/software versions, and antimalware software. The framework will also conduct an analysis based on database/repository information to determine the most likely and most dangerous cyber threats facing the SoS in the configuration it recommends for the mission and resilience strategy.
- *Human Agent Assessment:* The framework will recommend necessary privileged access settings for human agents across the SoS, while providing a recommendation for automated response versus automated attack absorption and delay while awaiting human input.

- *Cost–Benefit Analysis:* Finally, the framework would deliver a cost analysis for the proposed resilience strategies. This requires analytically leverage costing concepts, such as weighing the "annual loss expectancy" without security investment versus expected losses with the security investment [39]. This would allow the framework to determine expected loss due to information security compromises (both in terms of performance attributes and monetary cost). It would then weigh this against expected losses in light of the resilience strategy, numerically showing which alternative is preferable. When weighing alternatives, Dr. Jairus Hihn cautions to remember that the "Do Nothing Option" is a possibility [40]. If this is a hastily established SoS for a very near-term mission, the time needed to add a thorough resilience strategy may negate its operational value. In that case, the benefit of getting the SoS fielded quickly outweighs the warnings from the CVA.

12.5 Conclusion

Cyberspace is an inseparable reality of any DoD SoS, requiring militaries to defend their respective cyber enclave. And, just as the scope of the modern CP-SoS has become increasingly complicated and complex, so too have the means to architect resilience into the CP-SoS. This paper has presented a framework for developing cyber-resilient strategies within a model-based paradigm. The framework provided represents a basis for an architecting model to navigate the available resilience methods. Ultimately, each resilience strategy will represent a trade-off that best supports the priorities of those architecting a DoD SoS. However, the goal remains the same: the cyber-resilience strategy prevents "'drift' towards system brittleness, a harbinger of potential [failures]" [7]. The cyber-resilient SoS will manage its drift in the face of disruptions, flexing and adapting as necessary to prevent total failure while retaining a feasible pathway to the desired SoS objective. The challenge comes in managing resilience strategies, as the strategies change as the SoS and its context changes. What is resilient for a static forward operating base may not benefit a brigade combat team, may not be cost-effective for a quick reaction force, or may be impossible for a platoon. Thus, not only must the framework for resilience model the appropriate echelon, but it must also nest subordinate strategies as higher and lower abstraction-levels within the SoS are architected. Furthermore, the framework needs to be extended and augmented with appropriate data analytics and simulation capability to support trade-space exploration and multiobjective decision-making. Fortunately, the framework itself is adaptable and leverages diverse datasets. The art of architecting cyber-resilience, then, rests upon use of the framework to continually manage trade-offs effectively. This coupled with rolling CVAs and a robust CSA will ensure a continuous, resilient cyber-architecture for the DoD SoS of interest.

References

1. Cebrowski AK, Garstka JJ (1998) Network-centric warfare: its origin and future. US Naval Institute Proceedings 124(1):28–35
2. Office of the Deputy Under Secretary of Defense for Acquisition and Technology, Systems and Software Engineering (2008) Systems engineering guide for systems of systems, 1st edn. DoD, Washington, DC
3. Department of Defense (2013) Joint publication 3–12 (R): cyberspace operations. DoD, Washington, DC
4. Wilson C (2004) Network centric warfare: background and oversight issues for congress. The Library of Congress Congressional Research Service, Washington, DC
5. Singer PW, Friedman A (2014) Cybersecurity and cyberwar: what everyone needs to know. Oxford University Press, New York
6. Meeuwisse R (2015) Cybersecurity for beginners. Icurtain Ltd., Kent
7. Madni AM, Jackson S (2009) Towards a conceptual framework for resilience engineering. Syst J IEEE 3(2):181–191
8. Department of Defense (2015) The DoD cyber strategy. DoD, Washington, DC
9. Stradley J, Karraker D (2006) The electronic part supply chain and risks of counterfeit parts in defense applications. Comp Pack Technol IEEE Trans 29(3):703–705
10. IEEE Spectrum (2008) The hunt for the kill switch [Online]. Available: http://spectrum.ieee.org/semiconductors/ design/the-hunt-for-the-kill-switch. Accessed on: 28 July 2016
11. Madni AM, Sievers M (2014) System of systems integration: key considerations and challenges. Syst Eng 17(3):330–347
12. Rittel HWJ, Webber MM (1973) Dilemmas in a general theory of planning. Policy Sci 4 (2):155–169
13. Donaldson SE, Siegel SG, Williams CK, Aslam A (2015) Defining the cybersecurity challenge. In: Enterprise cybsersecurity. Apress, New York City
14. McAfee (2016) McAfee knowledge center [Online]. Available: https://kc.mcafee.com/corporate/index?page =content&id=KB55986. Accessed on: 15 June 2016
15. Reason J (2000) Human error: models and management. BMJ 320(7237):768–770
16. Neches R (2011) Engineered Resilient Systems (ERS) S&T Priority Description and Roadman
17. Georger SR, Madni AM, Eslinger OJ (2014) Engineered resilient systems: a dod perspective. Procedia Comput Sci 28:865–872
18. Madni (unpublished) Overview/spring 2016 – Lecture 1
19. Jabbour K, Poisson J (2016) Cyber risk assessment in distributed information systems. Cyber Def Rev 1(1):91–112
20. Hura M et al (2000) Interoperability: a continuing challenge in coalition air operations. RAND Corporation, Santa Monica, CA, Rep. MR-1235-AF
21. Harris S (2014) @War: the rise of the military-internet complex. Houghton Mifflin Harchour, New York
22. Jabbour K, Muccio S (2011) The science of mission assurance. J Strateg Secur 4(2):61
23. Australian Signals Directorate (2014) Top four mitigation strategies to protect your ICT system [Online]. Available: http://www.asd.gov.au/publications/protect/Top_4_Mitigations.pdf. Accessed on: 24 Nov 2015
24. Australian Signals Directorate (2014) Strategies to mitigate targeted cyber intrusions [Online]. Available: http://www.asd.gov.au/publications/Mitigation_Strategies_2014.pdf. Accessed on: 24 Nov 2015
25. Goldman H, McQuaid R, Picciotto J (2011) Cyber resilience for mission assurance. In: Technologies for Homeland Security (HST), IEEE International Conference on, Waltham
26. Baker M, Manweiler J (2015) From nifty gadgets to dire warnings. IEEE Pervasive Comput 4:6–11

27. Wired (2015) Hackers remotely kill a jeep on the highway – With me in it [Online]. Available: https://www.wired.com/2015/07/hackers-remotely-kill-jeep-highway/. Accessed on: 20 July 2016
28. Rechtin E, Architecting S (1991) Creating and building complex systems. Englewood Cliffs, Prentice Hall
29. Gaining the advantage: applying cyber kill chain ® methodology to network defense, Lockheed Martin, El Segundo, 2015
30. Donaldson S et al (2015) Enterprise cybersecurity: how to build a successful cyberdefense program against advanced threats. Apress, New York
31. DISA (2016) DISA – Assured Compliance Assessment Solution (ACAS) [Online]. Available: http://www.disa.mil/cybersecurity/network-defense/acas. Accessed on: 25 July 2016
32. Dasgupta D (2006) Advances in artificial immune systems. IEEE Comput Intell Mag 1 (4):40–49
33. Madni AZ, Madni CC, Salasin J (2002) 5.4. 1 proactTM: process-aware zero latency system for distributed, collaborative enterprises, In: INCOSE International Symposium, vol 12, no. 1
34. Schnjakin M, Meinel C (2013) Implementation of cloud-raid: a secure and reliable storage above the clouds. In: International Conference on Grid and Pervasive Computing, Seoul, Korea
35. Jules A, Oprea A (2013) New approaches to security and availability for cloud data. Commun ACM 56(2):64–73
36. Madni AM (2011) Integrating humans with and within complex systems. Cross Talk 24(3):4–8
37. Matthews ED, Arata HJ III, Hale BL (2016) Cyber situational awareness. Cyber Def J 1 (1):35–45
38. Nurse JRC et al (2014) Understanding insider threat: a framework for characterising attacks. In: Security and Privacy Workshops (SPW), 2014 IEEE, San Jose
39. Bohme R, Moore T (2012) Security metrics and security investment. Lyle School of Engineering, Southern Methodist University, Dallas, TX, Rep., 9 September
40. Hihn J (unpublished) Lecture 3: present worth & annual equivalence

Chapter 13
Inference Enterprise Multimodeling for Insider Threat Detection Systems

Edward Huang, Abbas K. Zaidi, and Kathryn B. Laskey

Abstract Organizations employ a suite of analytical models to solve complex decision problems in their respective domains. The current practice uses different simulation and modeling formalisms and subject matter experts to address parts of a larger problem. There is a realization that complex problems cannot be solved by employing a single analytical methodology and its supporting tools; rather, they require a combination of several such methods, all supplementing or complementing each other. We propose the use of multiformalism-based modeling and analysis to assist in evaluating performance of insider threat detection systems. The paper proposes a multimodeling test bed that allows integration of multiple modeling and analysis techniques that can digest and correlate different sources of data and provide insights on performance of insider threat detection systems.

Keywords Multimodeling • Insider threat detection • Inference enterprise

13.1 Introduction

The insider threat is manifested when user behavior departs from normal compliance with established processes and policies. The motive behind this behavior may be malice or a disregard for established policies in an organization. Reliance on computers for information storage and processing, and advances in networking technology for information access/exchange have created new means and venues, and in some cases incentives, for user behaviors outside the prescribed norms. The resulting new technology challenges have made the Insider Threat Detection (ITD) problem even harder, especially when the threat comes from technologically savvy individuals. Various approaches, countermeasures, and techniques have been proposed and used to mitigate the issue, including security policies, procedures, technical controls, indicators, and detection tools. Surveys (e.g., [1, 2]) provide a classification of existing and proposed intrusion and anomaly detection systems

E. Huang (✉) • A.K. Zaidi • K.B. Laskey
Systems Engineering & Operations Research Department, George Mason University, Fairfax, VA 22030, USA
e-mail: chuang10@gmu.edu; szaidi2@gmu.edu; klaskey@gmu.edu

A.M. Madni et al. (eds.), *Disciplinary Convergence in Systems Engineering Research*, https://doi.org/10.1007/978-3-319-62217-0_13

based upon the methodology used in their development. Statistical methods, information theoretic methods, Bayesian networks, game theoretic methods, principal component analysis, Markov processes and hidden Markov models, data mining methods (classification-/clustering-based), decision trees, and rule-based (production-/logic-based) methods are all examples of major methodologies that have been or could be used in developing IDT systems. A classification on the basis of network versus nonnetwork-based systems is also presented in [1]. Both [1, 2] cite several example application tools that are based on these methods. Greitzer et al. [3] provide categories of input data sources, both technical and social, that are used or required by the detection algorithms.

Due to the variety of threats and complexity of solution methods, an organization requires employment of a wide range of procedures and technical tools since a single procedure or a single detection tool may not be enough for all types of threats and anomalies. The term 'Inference Enterprise (IE)' is used to refer to the collection of data, tools, and algorithms that an organization employs to address IDT. Gritzalis et al. [4] argue that an IE must be studied in the context of an employing organization's business process model: an IE employed by one organization may not perform as well when employed by another organization due to differences in their business processes. They propose built-in security mechanisms that are able to detect and fight threats that manifest at different stages in the business process. In this paper, we are specifically concerned with the automated portion of an IE. We do note that real IEs also involve manual and semiautomated processes. Although we do not focus on modeling human systems, the models will necessarily include interfaces to human systems, and performance analysis may require assumptions about, and sensitivity analysis regarding, parameters characterizing processes performed by humans. This definition of an IE presents us with a domain that can be characterized with multiple data representations and multiple formalisms for inference, all linked together in an underlying business process model. The algorithms in an IE are designed to look for a certain pattern of user behavior, defined with the help of some combination of indicators, and measured via observables called detectors (see Fig. 13.1).

A holistic, integrated model of an IE, that is, IEM, for the purpose of evaluating its performance and effectiveness for specific scenarios of interest (i.e., input models for user population behaviors), therefore, requires a modeling framework that incorporates multiple data sources and modeling approaches in a semantically verifiable manner. The multimodel Semantic Test bed for Inference Enterprise Modeling (STIEM) is intended to address this challenge. STIEM can be employed to develop an Inference Enterprise Model (IEM) of a given inference enterprise. Because the overall inference process involves interoperations among multiple entities of the enterprise, an IEM is formalized with the help of a workflow language. A workflow implementation platform is used to represent a given inference enterprise to evaluate its performance for a given inference task. A workflow in our approach can be instantiated with multiple data sources and different parameter values of the detection algorithms. The workflow is automated for simulating the given enterprise model.

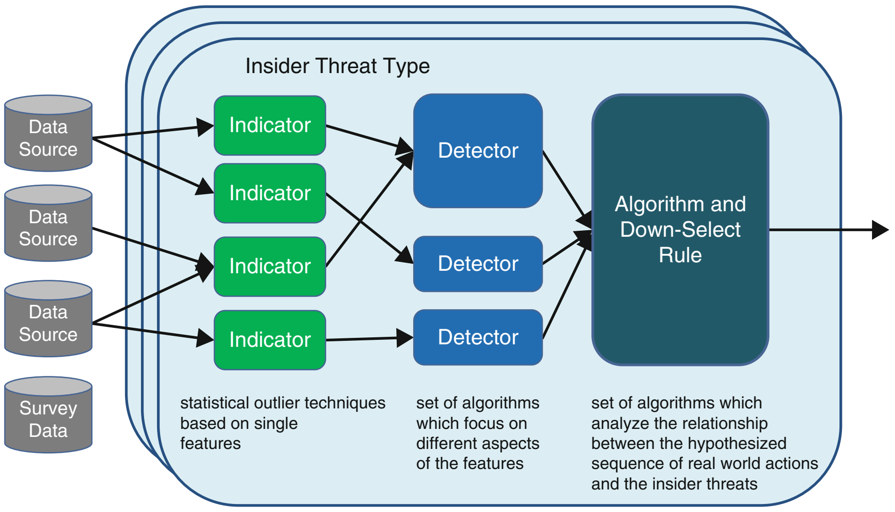

Insider Threat Type
Data Source
Data Source
Data Source
Survey Data
Indicator
Indicator
Indicator
Indicator
Detector
Detector
Detector
Algorithm and Down-Select Rule
statistical outlier techniques based on single features
set of algorithms which focus on different aspects of the features
set of algorithms which analyze the relationship between the hypothesized sequence of real world actions and the insider threats

Fig. 13.1 An inference enterprise

The recent research on multimodeling is based on the observation that computational models, created using different modeling techniques, usually serve different purposes and provide unique insights. While each modeling technique might be capable of answering specific questions, complex problems require multiple models interoperating to complement/supplement each other; we call this multimodeling. This multimodeling approach for solving complex problems is full of syntactic and semantic challenges. In [5], a theoretical foundation for the use of multiple interacting modalities in order to determine the valid interaction between different modeling techniques was presented. The proposed approach was based on the use of concept maps, meta-models, and ontologies to capture the valid interoperations between interconnected models. It extended earlier research efforts by Kappel et al. [6] and Saeki and Kaiya [7, 8]. A systematic, domain-specific methodology for addressing multimodeling problems is presented in [8–10]. This methodology has been illustrated by creating workflows of model interoperations involving Social Networks, Timed Influence Nets, Organization Structures, and Geospatial models in a variety of problem domains [11–14].

This paper is organized as follows. In Sect. 13.2, we introduce the multimodel Semantic Test bed for Inference Enterprise Modeling (STIEM). In Sect. 13.3, we illustrate the use of STIEM to develop and exercise a model of an insider threat detection system. Section 13.4 presents a summary and some conclusions and future work we can draw from this research.

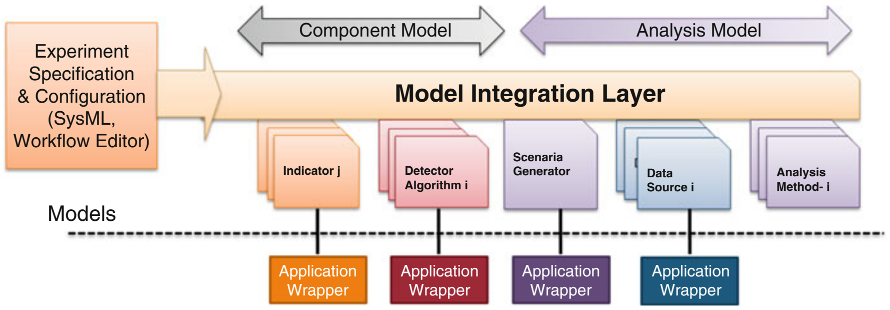


Fig. 13.2 STIEM architecture

13.2 Multimodel Semantic Test Bed for Inference Enterprise Modeling (STIEM)

The multimodel Semantic Test bed for Inference Enterprise Modeling (STIEM) is a computational framework that demonstrates the value of multiple integrated formalisms by serving as a platform for semantic and syntactic integration of multiple decision-making and inference models to form a model of an IE. STIEM supports an experimental environment for evaluating effectiveness of Inference Enterprises. STIEM) allows the modeling and simulation of a dynamic and adaptive Inference Enterprise (IE) using a systems engineering approach by incorporating data sources, indicators, detectors, algorithms, design and analysis tools on a configurable platform (Fig. 13.2). This test bed can be used for experimental studies that bring together different modeling, distributed computational infrastructure development, reasoning, decision support, and evaluation tools into an integrated environment for design, analysis, and performance studies. Some of the modeling formalisms that are available on STIEM include Bayesian networks, dynamic Bayesian networks, stochastic optimizers, discrete event systems, decision trees, rule-based systems, and others to be determined as the research progresses. This platform provides a repository of models of data sources, indicators, detector algorithms, and analysis processes and tools. The models in the repository are separately developed at the level of detail required to address a decision problem. These models are constructed using different formalisms and software applications with their specific data needs. The repository items are made available to the Model Integration Layer via application-specific wrappers developed at the Instrumentation Layer. A front-end workflow editor of STIEM allows IEM builders an ability to plug-and-play the repository data sources and algorithms in some specific partial order, or workflow. The front-end of STIEM allows for configuration changes to parameter values of workflow entities and supports performance of multiple simulation runs. The resulting experimental data can be processed by a library of analysis tools for performance and effectiveness analyses.

The STIEM test bed is built on top of a COTS software application called ModelCenter® [15]. Phoenix Integration's ModelCenter® allows for simple integration of components using various software platforms. It uses a model-based engineering framework that provides a wide variety of tools and methods to encapsulate individual analysis or simulation models, store them as reusable components, and create simulation workflows with different data sources. The individual simulation and/or analysis components of a workflow can be developed using any software application, programming language, spreadsheet, analytical model, or database. ModelCenter's simulation workflows can be easily developed using an editor that allows linking of reusable components (stored in a library) in a "building-block" approach. Once a workflow has been created, ModelCenter® can automatically execute the simulation workflow as many times as is needed. As the workflow executes, data are automatically transferred from one component to the next (across the network as necessary). Computationally expensive workflows can be executed using parallel computing resources to shorten run times. In addition, it contains a whole array of optimization, alternatives exploration, and visualization tools.

13.3 Application: Insider Threat Detection

This section illustrates the use of STIEM to develop a model of a fictitious insider threat detection system. Section 13.3.1 describes an insider threat detection system that attempts to identify employees who transfer data/information to other countries. Section 13.3.2 presents an optimization model used as a component of an IEM. In Sect. 13.3.3, we will create an analysis workflow including the optimization and simulation model and perform the sensitivity analysis of the workflow.

We assume that an automated insider threat detection system follows the following general process. First, a behavior is identified that characterizes a *threat type*. Second, one or more *indicators* are defined whose presence is associated with the threat type. Third, one or more *detectors* are defined to measure the presence of each indicator. Finally, an *alert* method is defined that combines detector outputs and applies a down-select rule to identify users whose behavior is to be investigated further. Such an automated insider threat detection system is in general imperfect. That is, there are users who match the threat type but whose behavior does not result in an alert; likewise, there are users whose behavior triggers an alert, but who do not match the threat type. Key objectives of an IEM are to evaluate how well the enterprise's automated threat detection system performs at detecting threats, to understand the reasons for its performance, and to identify ways to improve performance.

13.3.1 Example of an Insider Detection System

Consider a detection system used to find insiders who transfer data/information to other countries. Their associated behavior is to transfer files. To identify these behaviors, we can define two *indicators*: (i) the number of files transferred internationally; and (ii) the size of files sent internationally. The automated portion of the IE uses software tools to log file transfers. For each indicator, we define a detector for the associated behavior by defining a threshold, which if exceeded, triggers the detector. Finally, the IE applies a down-select rule, for example, that both detectors identify the same employee, to define alerts, that is, employees whose file transfer behavior merits investigation.

13.3.2 One Inference Enterprise Component: Optimization Models

To analyze the performance of an IE, various performance metrics are defined. For example, we might be interested in studying the false positive rate (percentage of alerts that do not match the threat type) and false negative rate (percentage of nonalerts that match the threat type). To study these quantities, we define a joint probability distribution on the factors of interest: whether the user matches the threat type, the indicators, the detectors, and whether an alert is issued.

For our fictitious inference enterprise, we define $P(T, I1, I2, D1, D2, A)$ as the joint probability of six Boolean (true/false) random variables: the threat, the two indicators, the two detectors, and the alert. We have defined an alert as a known deterministic function of detector values: $P(A|D1, D2)$ is true if D1 and D2 are both true and false otherwise. Thus, we need to estimate the joint distribution of the remaining variables: $P(T, I1, I2, D1, D2)$. Once we have this joint distribution, we can use it to calculate performance metrics of interest, such as the false positive and false negative rates.

We assume the joint distribution is estimated using data from past performance of the IE. One common approach to estimating a joint distribution from data is maximum likelihood estimation – we find the joint distribution that maximizes the likelihood of the observed data. Several issues arise in this connection. First, the full joint distribution may be intractable when there are many indicators and detectors. Second, many of the behaviors we are trying to detect are rare, raising the problem of estimating a probability from very few positive instances. Third, the threat variable and some of the indicators may not have a definite "ground truth," being measured only subjectively. When this is the case, there may be little to no prior data for estimating the model. Finally, full data may be unavailable, and so the models must be estimated from summary data.

A common approach to estimating a probability distribution from data is maximum likelihood. To apply the maximum likelihood method, we define a functional

Fig. 13.3 Probability table

Threat	I1	I2	D1	D2	P
TRUE	TRUE	TRUE	TRUE	TRUE	p[1]
TRUE	TRUE	TRUE	TRUE	FALSE	p[2]
TRUE	TRUE	TRUE	FALSE	TRUE	p[3]
TRUE	TRUE	TRUE	FALSE	FALSE	p[4]
TRUE	TRUE	FALSE	TRUE	TRUE	p[5]
TRUE	TRUE	FALSE	TRUE	FALSE	p[6]
TRUE	TRUE	FALSE	FALSE	TRUE	p[7]
TRUE	TRUE	FALSE	FALSE	FALSE	p[8]
...					
FALSE	FALSE	FALSE	FALSE	FALSE	p[32]

form for the distribution and then estimate its parameters by finding the parameters for which the likelihood of the data is maximized. The maximum likelihood estimation methods from many common statistical packages expect input in the form of a data set of values for all the random variables. Because we have only summary statistics, we took the approach of formulating the likelihood maximization problem as a nonlinear program and applying an off-the-shelf solver to find a joint probability distribution that maximizes the log-likelihood.

As shown in Fig. 13.3, we decoded every combination of values as one variable in the nonlinear optimization model. These variables represent the probabilities of their associated combination. For example, P3 represents the probability that the user matches the threat type and all indicators and detectors except Detector 1 are true.

We assume that the data provided are in the form of 2 × 2 contingency tables containing counts of pairs of random variables. Each pair of random variables has four possible values: TT, TF, FT, and FF. We assume the observations are independent and identically distributed, with respective probabilities $p_{\mathrm{TT}}, p_{\mathrm{TF}}, p_{\mathrm{FT}}$ and p_{FF}. If we observe counts $N_{\mathrm{TT}}, N_{\mathrm{TF}}, N_{\mathrm{FT}}$, and N_{FF} under this model, the log-likelihood of the 2 × 2 table is $(N_{\mathrm{TT}} * \log(p_{\mathrm{TT}}) + N_{\mathrm{TF}} * \log(p_{\mathrm{TF}}) + N_{\mathrm{FT}} * \log(p_{\mathrm{FT}}) + N_{\mathrm{FF}} * \log(p_{\mathrm{FF}}))$.

If all the 2 × 2 tables are mutually consistent, the log-likelihood function is maximized by finding a joint distribution in which each marginal 2 × 2 cell probability p_{TF} is equal to the respective data frequency. On the other hand, different 2 × 2 tables may come from different sources (e.g., computer logs, judgment of subject-matter experts), may have different levels of fidelity to "ground truth," and may be inconsistent with each other. We thus provide for the ability to specify a weight for each 2 × 2 table, indicating how much that table should contribute to the optimization.

We formulate our optimization problem as a nonlinear programming model. Each summary 2 × 2 data table is associated with a given weight parameter. The weight parameter, w_i, is the weight for Table i. w_i is a user-defined parameter between 0 and 10. If the table is not used at all, its weight is 0. If we fully trust the

Fig. 13.4 Example of the data tables

		I1	
		TRUE	FALSE
Threat	TRUE	211	63
	FALSE	1	3529

		I2	
		TRUE	FALSE
Threat	TRUE	211	63
	FALSE	1	3529

		I2	
		TRUE	FALSE
I1	TRUE	31	184
	FALSE	183	3869

table, for example, it corresponds to ground truth, we assign it a weight of 10. The user can assign any value between 0 and 10.

Denote the data by N and the number of tables by n. The problem formulation is shown as follows.

Obj.	$\text{Max}\sum_{i=1}^{n} w_i * (N_{\text{TT}} * \log(p_{\text{TT}}) + N_{\text{TF}} * \log(p_{\text{TF}}) + N_{\text{FT}} * \log(p_{\text{FT}}) + N_{\text{FF}} * \log(p_{\text{FF}}))$
s.t.	$p_j \geq 0\,, \forall j$
	$\sum p_j = 1\,, \forall j$

The objective function is to maximize the log-likelihood function. The first constraint is the nonnegative probability constraint and the second one is to enforce that the sum of all probabilities is equal to 100%.

To illustrate how the nonlinear program is defined, suppose we are considering only three random variables: threat, indicator 1, and indicator 2. Then, there are only eight variables in the optimization (parameters to be estimated).

Assume we are given data tables for Threat × $I3$, Threat × $I4$, and $I3$ × $I4$, as shown in Fig. 13.4.

The first three columns of Fig. 13.5 show the eight configurations of the Threat, $I3$, and $I4$ variables.

The objective function and constraints for the corresponding NLP are given as follows.

$Obj.Function$:

$$31*\log(p[1]+p[5]) + 184*\log(p[2]+p[6]) + 183*\log(p[3]+p[7]) + 3869*\log(p[4]+p[8]) + w*\left(211*\log(p[1]+p[2]) + 63*\log(p[3]+p[4]) + \log(p[5]+p[6]) + 3529*\log(p[7]+p[8]) + w*(211*\log(p[1]+p[3]) + 63*\log(p[2]+p[4]) + \log(p[5]+p[7]) + 3529*\log(p[6]+p[8])\right)$$

$$\text{s.t.}\, p[i] \geq 0, i = 1 \cdots 8 \sum_{i=1}^{8} p[i] = 1$$

Fig. 13.5 Example of the NLP encoding process

Threat	I1	I2	p[i]	NLP
TRUE	TRUE	TRUE	p[1]	0.0118
TRUE	TRUE	FALSE	p[2]	0.0338
TRUE	FALSE	TRUE	p[3]	0.0337
TRUE	FALSE	FALSE	p[4]	0.000001
FALSE	TRUE	TRUE	p[5]	0.000001
FALSE	TRUE	FALSE	p[6]	0.000377
FALSE	FALSE	TRUE	p[7]	0.000375
FALSE	FALSE	FALSE	p[8]	0.92

The last column of Fig. 13.5 shows the NLP solution: the probabilities that maximize the objective function subject to the constraints.

13.3.3 Analysis Workflow

An analysis workflow for modeling the example insider threat enterprise is shown in the following figure. We develop the workflow in the tool, ModelCenter™. The first step, AMPL Wrapper, is to run the optimization model described in Sect. 13.3.2, using the AMPL solver. It will use the weights that user assigned, create the formulation, run the AMPL nonlinear solver and output the probabilities of all combinations. The result of the nonlinear solver, that is, the probability of each combination, will be the input to the simulation model. The simulation model, written in Java, simulates a user population using the probabilities provided from the AMPL solver, and calculates various performance metrics (e.g., false positive and false negative rates) for the simulated enterprise. This simulator module is shown as javaSimulatorWrapper in Fig. 13.6. We ran the simulation 1000 times, generating performance metrics for 1000 simulated enterprises. These results are passed to the StatCalculate step, where means, standard deviations, and percentiles are calculated. The last step, that is, Excel, is used to collect the results and produce tables and plots.

The analysis workflow implemented in ModelCenter ™ then can enable us to perform sensitivity analysis on the model. For example, in Fig. 13.7, we study the weights (W1 and W2) of the optimization model. As shown in the figure, the false positive rate of the whole analysis workflow will significantly depend on W1. When W1 is less than 1, the false positive rate is more than 3.5%. Otherwise, the result is less than 3.5%.

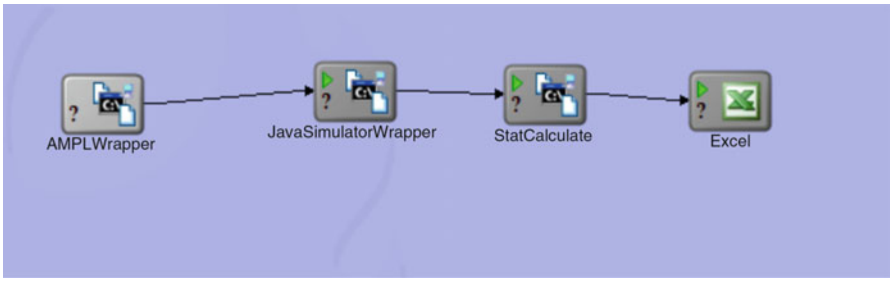


Fig. 13.6 The process of the stochastic optimization approach (implemented in ModelCenter™)

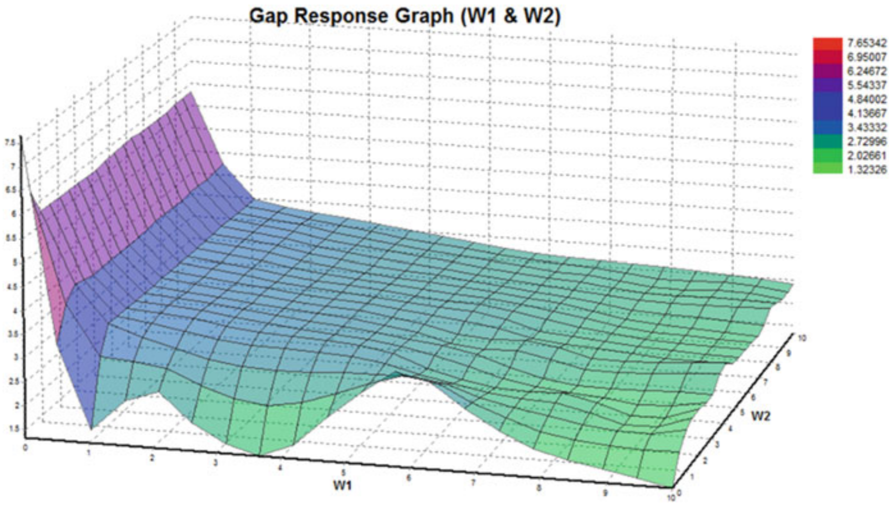


Fig. 13.7 Sensitivity analysis of an insider threat detection example

13.4 Conclusion

Insider threat detection is a difficult and highly challenging problem. There is a wide range of intelligent threats, each using unique and innovative ways to avoid detection, each requiring different detection and mitigation strategies. Constant innovation of threats contributes to weak detectors that result in high false positive rates when tuned to give a reasonable level of detection. Avoidance, detection, and mitigation processes are currently implemented with little formal test and evaluation.

The modeling and analysis test bed presented in this paper addresses the following research objectives:

1. In the absence of formal analytical models giving us closed-form answers to our security questions, use engineering modeling within an experimental test bed where existing and proposed IEs can be tested and vetted.

2. Use empirical studies to develop an understanding of the performance of an IE. Develop a quantitative measure of fitness of an IE for a given organization's needs.
3. Provide a capability for risk and cost–benefit analysis for the alternative solutions.

Current practice involves largely manual design of IEs, a primarily reactive approach to fixing problems with little or no ability to proactively anticipate problems and predict future threat patterns.

Our research provides the following innovations that address the above research objectives:

- Mathematically rigorous, semantically meaningful framework for integration and interoperation of models developed under multiple modeling paradigms, using disparate data sources, and implemented in different computational tools
- Test bed enabling experimentation, evaluation, and reuse of models
- Enterprise modeling paradigm that allows explicit representation of uncertainty about the structure and processes of an enterprise, providing rigorous and well-justified uncertainty bounds
- An experimental platform to study trade-offs between complex, high-fidelity models with many adjustable parameters against simpler, lower-fidelity models with fewer parameters
- A platform for performing validation and verification of different Inference Enterprise designs

The research reported here takes us a step closer to having the ability to design an inference enterprise with the best capability to mitigate a class of threats without impeding legitimate activities. In order to improve the performance of an IE, we must be able to measure its performance, predict how it will perform, understand the reasons why it performs as it does, and predict how proposed changes will affect its performance. The purpose of our research is to provide a way to evaluate proposed architectures with respect to their ability to identify genuine threats without hindering legitimate behavior.

Acknowledgments Research reported here was supported under IARPA contract 2016 16031400006. The content is solely the responsibility of the authors and does not necessarily represent the official views of the US Government.

References

1. Patcha A, Park JM (2007) An overview of anomaly detection techniques: existing solutions and latest technological trends. Comput Netw 51(12):3448–3470
2. Chandola V, Banerjee A, Kumar V (2009) Anomaly detection: a survey. ACM Comput Surv 41(3):15

3. Greitzer FL, Paulson P, Kangas L, Franklin LR, Edgar TW, Frincke DA (2008) Predictive modelling for insider threat mitigation. Pacific Northwest National Laboratory, Richland, WA, Tech. Rep. PNNL Technical Report PNNL-60737
4. Gritzalis D, Starvou V, Kandias M (2014) Insider threat: enhancing BPM through social media, 2014 6th International Conference on New Technologies, Mobility and Security (NTMS)
5. Levis AH, Zaidi AK, Rafi MR (2012) Multi-modeling and meta-modeling of human organizations. In: Salvendy G, Karwowski W (eds) Advances in human factors and ergonomics 2012: Proceedings of the 4th AHFE conference 21–25 July 2012, CRC Press
6. Kappel G, Kapsammer E, Kargl H, Kramler G, Reiter T, Retschitzegger W, Schwinger W, Wimmer M (2006) Lifting metamodels to ontologies: a step to the semantic integration of modeling languages. In: International conference on model driven engineering languages and systems. Springer, Berlin Heidelberg, pp 528–542
7. Saeki M, Kaiya H (2006) On relationships among models, meta models and ontologies. In: Proceedings of the 6th OOPSLA Workshop on Domain-Specific Modeling (DSM 2006)
8. Jbara AA On using meta-modeling and multi-modeling to address complex problems (Doctoral dissertation, George Mason University)
9. Jbara AA, Levis AH, Zaidi AK (2013) On using multiple interoperating models to address complex problems. In: Proceedings Conference on Systems Engineering Research, CSER 2013, Atlanta
10. Levis AH, Jbara AA (2013) Multi-modeling, meta-modeling and workflow languages. In: Gribaudo M, Iacono M (eds) Theory and application of multi-formalism modeling. IGI Global, Hershey
11. Mansoor F, Zaidi AK, Wagenhals L, Levis AH (2009) Meta-modeling the cultural behavior using timed influence nets. In: Second international workshop on social computing, behavior modeling and prediction, Phoenix, AZ, April, 2009
12. Mansoor F, Zaidi AK, Wagenhals L, Levis AH (2009) Meta-modeling the cultural behavior using timed influence nets. In: Liu H, Salerno JL, Young MJ (eds) Social computing, behavioral modeling and prediction. Springer
13. Mansoor F, Zaidi AK, Levis AH (2010) Meta-model driven construction of timed influence nets. In: Proc. Workshop on Current Issues in Predictive Approaches to Intelligence and Security Analytics, IEEE Intelligence and Security Informatics Conference (ISI 2010), Vancouver, BC, Canada, May 2010
14. Levis AH, Wagenhals LW, Zaidi AK (2010) Multi-modeling of adversary behaviors. In: Proc. Workshop on Current Issues in Predictive Approaches to Intelligence and Security Analytics, IEEE Intelligence and Security Informatics Conference (ISI 2010), Vancouver, BC, Caneada, May 2010
15. Model Center, http://www.phoenix-int.com/modelcenter/integrate.php

Chapter 14
SoS Explorer: A Tool for System-of-Systems Architecting

David M. Curry and Cihan H. Dagli

Abstract System-of-systems (SoS) architecting is an important and difficult problem. Modeling and optimization are practically essential to develop a quality solution. However, modeling and optimization are highly specialized fields in their own rights and can easily become an obstacle to the architecting effort. SoS Explorer is a tool designed to mitigate this difficulty by providing a structured, yet flexible approach. Moreover, SoS explorer provides interactive visualization as well as a number of optimizers. Interactive visualization allows the architect to perform "what-if" analysis while the optimizers provide solutions that can act as initial architectures and demonstrate the optimal trade-space. The utility of this approach is demonstrated with a notional 22-system toy problem.

Keywords System of systems • SoS • architecting • Many-objective optimization problem • MaOP

14.1 Introduction

System of systems (SoS) are important to the functioning of modern society. They define infrastructures such as transportation, energy, and healthcare. They are used to achieve specific needs such as military missions as well as future plans for an intelligent transportation system or smart cities. As important as they are, SoS architecting remains a difficult problem due to the interplay of a large number of variables. Modeling can greatly benefit this effort; however, modeling is itself a difficult task. Deciding upon a modeling approach and implementing it along with the requisite models is not trivial and can consume significant time and resources. A tool for SoS architecting that already implements a flexible modeling approach can allow the architect to model the SoS while eliminating much of the difficulty. SoS Explorer is designed to be such a tool.

One way in which SoS Explorer provides value is by structuring the SoS architecting modeling effort. This structure helps to direct the model development

D.M. Curry (✉) • C.H. Dagli
Missouri University of Science and Technology, Rolla, MO, USA
e-mail: dmcfh3@mst.edu; dagli@mst.edu

A.M. Madni et al. (eds.), *Disciplinary Convergence in Systems Engineering Research*, https://doi.org/10.1007/978-3-319-62217-0_14

and to allow the tool to run and interpret the results allowing for interactive visualization, optimization, and negotiation. Interactive visualization allows the architect to perform "what-if" analysis via a graphical representation of the SoS architecture. Optimization methods give the architect a set of optimal architectures on which to base the final solution. The negotiation models provide insights into how the SoS may be affected by management decisions regarding resource allocation.

14.2 SoS Explorer

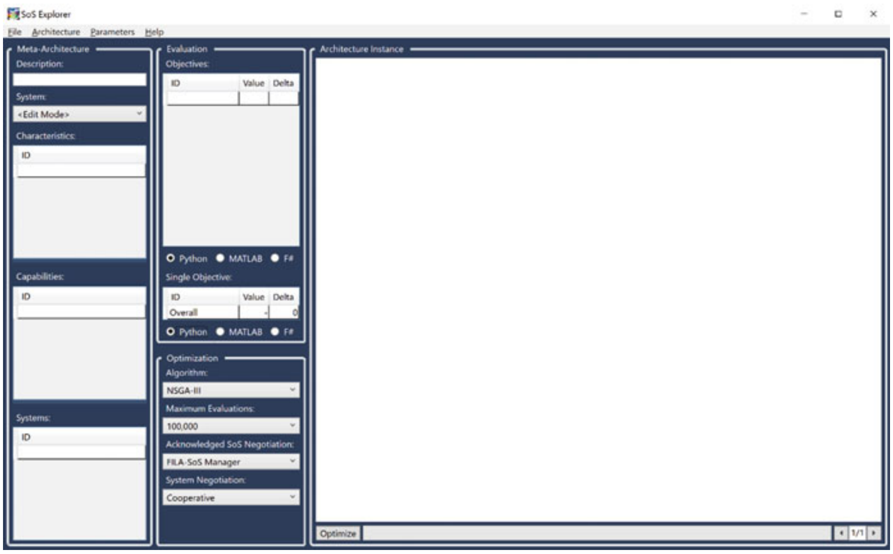
The SoS Explorer tool consists of four major components: problem definition, evaluation, optimization, and solutions. The graphical user interface is shown in Fig. 14.1 to illustrate how this is laid-out in the SoS Explorer. The purpose of this layout is to guide the architect through the steps of defining an architecture using this approach.

The purpose of the problem-definition section is to define a meta-architecture. The meta-architecture describes how individual architectures (or architecture instances) are composed. The meta-architecture allows for description of the problem to be entered and for the systems available for the SoS to be defined in terms of their characteristics, capabilities, and feasible interfaces. These are described in the SoS model.

The evaluation section is where the key performance measures (KPMs) are defined. These are referred to as objectives in deference to how they are used in optimization. The objectives are calculated based on the characteristics, capabilities, and feasible interfaces of the systems and interfaces selected in a given architecture. Therefore, the objectives define the modeling required for the SoS. The SoS Explorer allows the objectives to be calculated using the following languages: Python, MATLAB, or F#. An overall objective can also be defined that takes the other objectives as arguments. Its purpose is to enable single-objective optimization. For each objective, there is a "delta" as well as a calculated value. The delta shows the difference in the values between architectures or when an architecture is modified.

The optimization section allows the architect to choose an algorithm with which to find optimal architectures (solutions) and its termination criteria. There are two multiple-objective algorithms and one single-objective algorithm from which to choose. The algorithms can be tuned by modifying their parameters from the "Parameters" menu. The termination criteria are a combination of maximum number of evaluations and the point of detection of convergence. This is selected from the drop-down box. Negotiation modeling is planned but not yet implemented.

The solution (architecture instance) section represents the architecture solutions visually. These solutions are maintained as a set and may be paged through, added to, and modified. The architect can interact with the solutions by right-clicking on



System: <Edit Mode>

Single Objective: Overall

Algorithm: NSGA-III

Maximum Evaluations: 100,000

Acknowledged SoS Negotiation: FILA-SoS Manager

System Negotiation: Cooperative

Fig. 14.1 SoS Explorer graphical user interface

systems or interfaces and by left-clicking to create new interfaces. The problem and its solutions may then be saved as an Excel-format file.

14.3 SoS Model

A model should be as simple as possible for the purpose at hand and, at the same time, should not artificially restrict the problem being modeled. There are two key components of every SoS: systems and interfaces [1]. Therefore, systems and interfaces will be the basis for the SoS model. Keeping it simple, systems can either participate in the SoS or not. An interface either exists between two systems or it does not. Regarding the individual systems, they are characterized by the following: characteristics, capabilities, and feasible interfaces.

For the system model, characteristics are real-valued quantities and can represent items such as the cost of the system, the cost of implementing an interface with another system, performance measures, time to complete, etc. Capabilities are Boolean and represent individual capabilities of each system. These are sometimes referred to as the “little C” capability which contributes to the desired overall or the “big C” capability required by the SoS. The feasible interfaces are also Boolean and indicate whether it is possible to implement or have an interface between two systems.

The purpose of the system models is to estimate key performance measures (KPMs) for the SoS. The KPMs are used as the objectives while using an optimization method as well as to provide feedback on changes made interactively to an architecture. The only system characteristics, capabilities, and interfaces that need to be considered are those that affect the KPMs used to measure the performance of the SoS.

14.4 Optimization

The solution space of possible SoS architectures cannot be assumed to have a definable gradient because there is no reason for changes in architecture to produce continuous, let alone smooth, changes in their evaluation. Therefore, a non-gradient method must be employed for optimization. Evolutionary algorithms (EAs) are popular, actively researched non-gradient methods that can be readily applied to selection problems such as the given SoS model where systems and interfaces are selected to participate in the SoS. Using EAs to optimize the SoS architecture, the architecture must be represented as a chromosome. This is straightforward as each system's participation and each specified interface can be represented as a Boolean value. Therefore, the chromosome can be defined by $(n^2 + n)/2$ bits for undirected interfaces and n^2 bits for directed interfaces as shown in Fig. 14.2.

SoS Explorer supports both single- and multiple-objective optimizations. The user-defined objectives, O_i, are of the form $O_i : M_{\text{Char}}, M_{\text{Cap}}, M_{\text{Feas}}, C \mapsto \mathbb{R}, 1 \leq i \leq N$, where $M_{\text{Char}} \in \mathbb{R}^{n \times a}$ is the characteristics matrix, $M_{\text{Cap}} \in \{\text{T}, \text{F}\}^{n \times b}$ is the capabilities matrix, $M_{\text{Feas}} \in \{\text{T}, \text{F}\}^{n \times n}$ is the interface-feasibility matrix, $C \in \mathbb{R}^{\ell}$ is the chromosome, N is the number of objectives, n is the number of systems, a is the number of characteristics, b is the number of capabilities, and ℓ is the number of bits in the chromosome. For single-objective optimization, another function, O_{Overall}, of the form $O_{\text{Overall}} : O_1, O_2, \cdots, O_N \mapsto \mathbb{R}$ needs to be defined by the user.

The multiple-objective EAs included with the SoS Explorer are NSGA-III [2] and MOEA-DM [3]. Both of these are, more specifically, many-objective optimization (MaOP) algorithms. The typical SoS tends to have four or more objectives which creates issues for standard multi-objective approaches using Pareto dominance [4]. The many-objective approaches employ other criteria outside Pareto dominance to overcome these issues. The SoS Explorer comes with one single-

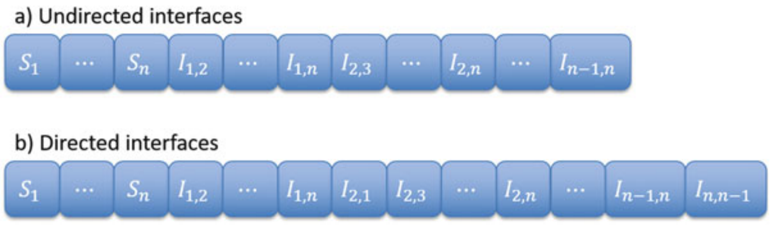

a) Undirected interfaces
S_1 … S_n $I_{1,2}$ … $I_{1,n}$ $I_{2,3}$ … $I_{2,n}$ … $I_{n-1,n}$
b) Directed interfaces
S_1 … S_n $I_{1,2}$ … $I_{1,n}$ $I_{2,1}$ $I_{2,3}$ … $I_{2,n}$ … $I_{n-1,n}$ $I_{n,n-1}$

Fig. 14.2 Chromosomes for SoS with (**a**) undirected and (**b**) directed interfaces

objective EA—Simple SOGA that is a plain multi-modal EA using single-point crossover and bit-flip mutation [5].

14.5 Demonstration

To demonstrate the use of the SoS Explorer, an example problem is presented. The problem is a notional intelligence, surveillance, and reconnaissance (ISR) problem consisting of 22 systems [6]. Each selected system contributes one or more of the following capabilities: electro-optical/infra-red (EO/IR), synthetic aperture radar (SAR), exploitation, command and control (C^2), and communication. The SoS must have all of these capabilities to be feasible. The individual systems are characterized by interface development cost, operational cost, performance, and development time. The KPMs (objectives) are performance, affordability, flexibility, and robustness. The objectives are modeled by Eqs. 14.1, 14.2, 14.3, and 14.4 respectively.

$$\text{Performance} = \sum\nolimits_{i=1}^{n} \left(\begin{cases} \text{Perf}_i, \text{if } S_i \\ 0, \text{otherwise} \end{cases} \right) \prod\nolimits_{j=1}^{n} \begin{cases} 1+\delta, \text{if } S_j \wedge I_{ij} \\ 1, \text{otherwise} \end{cases} \tag{14.1}$$

$$\text{Affordability} = -\sum\nolimits_{i=1}^{n} \left(\begin{cases} \text{Ops Cost}_i, \text{if } S_i \\ 0, \text{otherwise} \end{cases} \right) \sum\nolimits_{j=1}^{n} \begin{cases} I/F\ \text{Cost}_i, \text{if } I_{ij} \\ 0, \text{otherwise} \end{cases} \tag{14.2}$$

$$\text{Flexibility} = \sum\nolimits_{i=1}^{n} \sum\nolimits_{j=1}^{m} \begin{cases} 1, \text{if } S_i \wedge \text{Cap}_{ij} \\ 0, \text{otherwise} \end{cases} \tag{14.3}$$

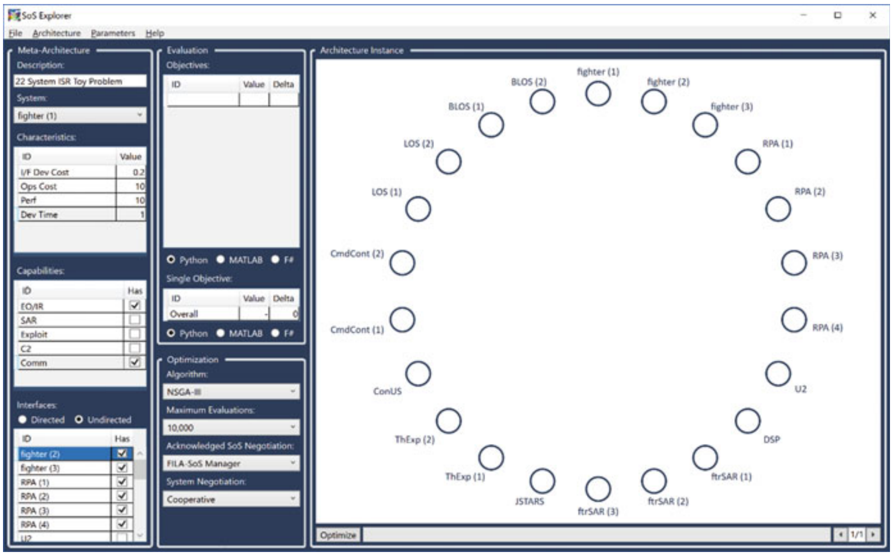
$$\text{Robustness} = -\max\left(\begin{cases} \text{Perf}_i, \text{if } S_i \\ 0, \text{otherwise} \end{cases}, 1 \le i \le n \right) \tag{14.4}$$

where $S_i, I_{ij}, \text{Perf}_i, \text{Ops Cost}_i, \text{I/F Cost}_i, \text{Cap}_{ij}$, and δ represent the ith system's participation, the interface between the ith and jth systems, ith system's performance, ith system's operational cost, ith system's interface cost, ith system's jth capability, and the performance boost provided by each implemented interface, respectively. When using an optimization algorithm, feasibility may be encouraged using a penalty function. The penalty function used in this case is

$$\text{Penalty} = -\sum\nolimits_{i=1}^{m} \begin{cases} \rho, \text{if missing Cap}_i \\ 0, \text{otherwise} \end{cases}$$

where ρ is the penalty for each missing capability in the SoS. The penalty is added to each objective, and ρ is chosen such that the penalty for infeasible solutions outweighs the actual objective.

To solve this problem in the SoS Explorer, the following steps may be performed:



Meta-Architecture

Description: 22 System ISR Toy Problem

System: fighter (1)

Characteristics:

ID	Value
I/F Dev Cost	0.2
Ops Cost	10
Perf	10
Dev Time	1

Capabilities:

ID	Has
EO/IR	✓
SAR	
Exploit	
C2	
Comm	✓

Interfaces: Directed [✓ Undirected]

ID	Has
fighter (2)	✓
fighter (3)	✓
RPA (1)	✓
RPA (2)	✓
RPA (3)	✓
RPA (4)	✓
U2	

Evaluation

Objectives:

ID	Value	Delta

[✓ Python] MATLAB F#

Single Objective:

ID	Value	Delta
Overall	-	0

[✓ Python] MATLAB F#

Optimization

Algorithm: NSGA-III

Maximum Evaluations: 10,000

Acknowledged SoS Negotiation: FILA-SoS Manager

System Negotiation: Cooperative

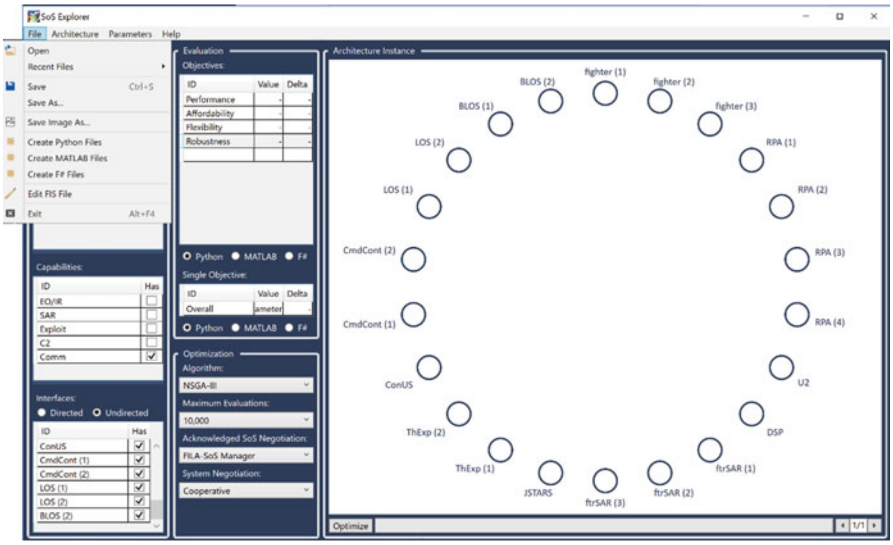
Fig. 14.3 SoS Explorer meta-architecture being defined

1. Enter the information for the meta-architecture: systems, characteristics, capabilities, and feasible interfaces.
2. Enter the names of the objectives.
3. Create the code templates in the desired languages using “File → Create Python/MATLAB/F# Files.”
4. Modify the resulting source files to implement the objective functions.

Step 1 is shown in Fig. 14.3 and steps 2–3 are shown in Fig. 14.4.

Now the ISR problem is modeled and architecture work can begin. A reasonable starting point would be to generate optimal solutions using one of the supplied methods. The results using Simple SOGA are shown in Fig. 14.5. To generate the set of optimal solutions, select an algorithm, termination criteria, and click on the “Optimize” button. The progress bar shows how much longer the calculation has to finish.

The individual solutions returned by the optimization algorithm can be accessed by paging through using the left and right arrows in the lower right-hand corner. The values in the objectives show the architecture’s assessment while the deltas show how the current architecture’s assessment changed from the previous architecture’s. Furthermore, any architecture may by modified by hand through the visual representation. By changing the systems’ participation and their interfaces, the architect can perform “what-if” analysis and design a custom solution using the feedback provided in the evaluation section.



Objectives:

ID	Value	Delta
Performance	-	-
Affordability	-	-
Flexibility	-	-
Robustness	-	-

Single Objective:

ID	Value	Delta
Overall	ameter	

Capabilities:

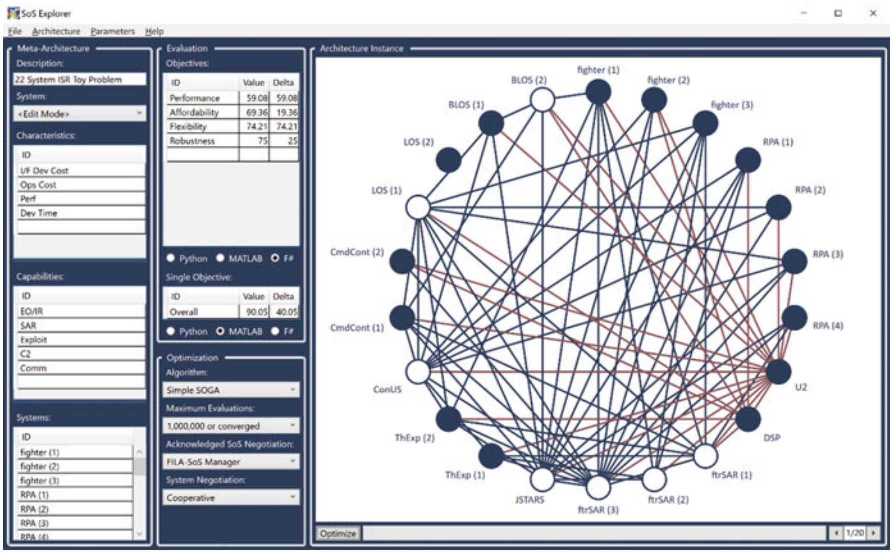
ID	Has
EO/IR	☐
SAR	☐
Exploit	☐
C2	☐
Comm	✓

Interfaces: Directed Undirected

ID	Has
ConUS	✓
CmdCont (1)	✓
CmdCont (2)	✓
LOS (1)	✓
LOS (2)	✓
BLOS (2)	✓

Optimization
Algorithm: NSGA-III
Maximum Evaluations: 10,000
Acknowledged SoS Negotiation: FILA-SoS Manager
System Negotiation: Cooperative

Fig. 14.4 SoS Explorer objectives named and code being created



Description: 22 System ISR Toy Problem
System: <Edit Mode>

Characteristics:

ID
U/F Dev Cost
Ops Cost
Perf
Dev Time

Capabilities:

ID
EO/IR
SAR
Exploit
C2
Comm

Systems:

ID
fighter (1)
fighter (2)
fighter (3)
RPA (1)
RPA (2)
RPA (3)
RPA (4)

Objectives:

ID	Value	Delta
Performance	59.08	59.08
Affordability	69.36	19.36
Flexibility	74.21	74.21
Robustness	75	25

Single Objective:

ID	Value	Delta
Overall	90.05	40.05

Optimization
Algorithm: Simple SOGA
Maximum Evaluations: 1,000,000 or converged
Acknowledged SoS Negotiation: FILA-SoS Manager
System Negotiation: Cooperative

Fig. 14.5 SoS Explorer solutions found by optimization algorithm

14.6 Conclusion

The SoS Explorer allows an architect to model, optimize, and visualize SoS architectures. The architect is provided a framework in which simple models allow architectures to be manipulated and evaluated. Optimization algorithms are provided that find candidate solutions for the architect. The architect can then compare and interact with these solutions to gain an understanding of the solution trade space. With this information, the architect can create a final solution while the feedback provided by the real-time architecture evaluation helps in guiding the design.

14.7 Future Work

A negotiation model is planned for the SoS Explorer but has not yet been implemented. The negotiation model could allow for competitive, semi-cooperative, and cooperative negotiation between the systems and a managing authority. A method such as chromosome fixing could be employed to enforce feasibility and provide constraints. Significant performance gains could be obtained by removing infeasible interfaces from the search space defined by the chromosome. If certain systems are required by the SoS, there should be a column where those systems could be so marked, and these systems should also be removed from the search space.

Appendix A. Python Source Code

The full Python code used in the ISR example for the "Performance" objective is listed below. Due to space constraints, the other objectives are not listed but follow similarly from their definitions. The code in lines 1–85 was automatically generated by SoS Explorer requiring only lines 86–103 (implementing Eq. 14.1) to be written by the user.

```
# Container class.
class Objective:

    # Class constructor
    def __init__(self):
        pass

    # Calculate the objective's value for a given architecture.
    def Objective_Performance(self, characteristics, capabilities,
                              feasibleInterfaces, architecture):

        # Delta bump for interfaces
        delta = 0.02

        # Bounds
        lowerBound =   0.0
        upperBound = 100.0
        bias       =   0.0

        # Number of elements
        numSystems         = 22
        numCharacteristics = 4
        numCapabilities    = 5

        # System indices
        sys_fighter1 = 0
        sys_fighter2 = 1
        sys_fighter3 = 2
        sys_RPA1     = 3
        sys_RPA2     = 4
        sys_RPA3     = 5
        sys_RPA4     = 6
        sys_U2       = 7
        sys_DSP      = 8
        sys_ftrSAR1  = 9
        sys_ftrSAR2  = 10
        sys_ftrSAR3  = 11
        sys_JSTARS   = 12
        sys_ThExp1   = 13
        sys_ThExp2   = 14
        sys_ConUS    = 15
        sys_CmdCont1 = 16
        sys_CmdCont2 = 17
        sys_LOS1     = 18
        sys_LOS2     = 19
        sys_BLOS1    = 20
        sys_BLOS2    = 21

        # Characteristic indices
        char_IFDevCost = 0
        char_OpsCost   = 1
        char_Perf      = 2
        char_DevTime   = 3

        # Capability indices
        cap_EOIR    = 0
        cap_SAR     = 1
        cap_Exploit = 2
        cap_C2      = 3
        cap_Comm    = 4

        # Interface type
        directedInterfaces = len(architecture) == numSystems * numSystems

        # Define hasSystem() function
        # Returns whether the given system is specified in the chromosome
        def hasSystem(i):
            # Return whether system is specified
            return architecture[i]

        # Define hasInterface() function
        # Returns whether the given interface is specified in the chromosome
        def hasInterface(i, j):
            # If same system
            if i == j:
                return False
            else:
                # Index of interface in chromosome
                jj = j + 1 if j < i else j
                m = min(i, j) + 1
                k = (numSystems - 1) * (i + 1) + jj if directedInterfaces else \
                    m * numSystems - m * (m - 1) / 2 + abs(i - j) - 1
                # Return whether interface is specified
                return architecture[k]

        # Calculate performance
        maxPerf = 0.0
        totalPerf = 0.0
        for i in range(0, numSystems):
            maxFeasible = 1.0
            totalFeasible = 1.0
            for j in range(0, numSystems):
                if feasibleInterfaces[i, j]:
                    maxFeasible *= (1.0 + delta)
                    if hasSystem(j) and hasInterface(i, j):
                        totalFeasible *= (1.0 + delta)
            maxPerf += maxFeasible * characteristics[i, char_Perf]
            if hasSystem(i):
                totalPerf += totalFeasible * characteristics[i, char_Perf]
        perf = lowerBound + (upperBound - lowerBound) * totalPerf / maxPerf

        # Return bounded results
        return max(lowerBound, min(upperBound, perf + bias))
```

References

1. Pape L, Giammarco K, Colombi J, Dagli C, Kilicay-Ergin N, Rebovich G (2013) A fuzzy evaluation method for system of systems meta-architectures. Procedia Comput Sci 16:245–254
2. Deb K, Jain H (2014) An Evolutionary many-objective optimization algorithm using reference-point-based Nondominated sorting approach, part I: solving problems with box constraints. IEEE Trans Evol Comput 18(4):577–601
3. Curry D, Tauritz D, Dagli C Many-objective evolutionary algorithm for decision makers. In preparation
4. Farina M, Amato P (2002) On the optimal solution definition for many-criteria optimization problems. In Fuzzy Information Processing Society, Proceedings NAFIPS 2002 Annual Meeting of the North American, pp 233–238
5. Eiben AE, Smith JE (2003) Introduction to Evolutionary Computing, ser. Natural computing. Springer, Berlin/Heidelberg
6. Dagli C et al (2015) Flexible and intelligent learning architectures for SoS (FILA-SoS), Volume 1 – Integrated Model Structure. Technical Report SERC-2015-TR-021-4, February

Chapter 15
A Principles Framework to Inform Defence SoSE Methodologies

Jaci M. Pratt and Stephen C. Cook

Abstract This paper is concerned with codifying the principles for successful system of systems engineering (SoSE) practice. The purpose of the codification is manifold but the initial focus is to support the design and utilization of system of systems engineering (SoSE) methodologies. The paper opens with a description of the problem context, defence capability engineering, and then moves on to describe an IDEF0 depiction of the inputs, controls and mechanisms needed to undertake a SoSE methodology design and utilization process; one of which is the set SoSE principles. Earlier work by the authors uncovered a substantial set of such principles and this paper concentrates on how to structure them to reduce the number needing consideration at any one time. The derivation of a three-layer framework designed to hold the principles follows. The framework comprises an articulation of the worldview which makes SoSE meaningful, the concepts that drive SoSE methodology design and use (descriptive heuristics) and implementation principles (prescriptive heuristics).The paper concludes with an outline of the content of the framework and suggestions for how it can be employed for its stated purpose.

Keywords System of systems • Framework • Defence • Methodologies

15.1 Introduction

This paper builds on earlier work by Cook et al. [11, 12] who gathered a substantial number of success factors for SoSE from the literature and illustrated how these could be applied to integrate a set of army capabilities sourced from a number of different capital equipment projects. Subsequent work more than doubled the original 48 principles and questioned the ability of the original structure to guide practitioners to salient principles for any given issue. Thus, a need arose to

J.M. Pratt (✉)
Defence Science and Technology Group, Third Ave, Edinburgh, SA 5111, Australia
e-mail: Jaci.Pratt@dsto.defence.gov.au

S.C. Cook
The University of Adelaide, Adelaide 5005, Australia
e-mail: Stephen.Cook@adelaide.edu.au

A.M. Madni et al. (eds.), *Disciplinary Convergence in Systems Engineering Research*, https://doi.org/10.1007/978-3-319-62217-0_15

restructure the principles into a richer hierarchy with a reduced number of principles that need to be considered at any one time [44]. This paper provides a consolidated and redrafted set of principles that are less descriptive and more directive to better inform the design and use of SoSE methodologies.

This work is being undertaken within the context of the recommendations of the recent Australian Defence First Principles Review (FPR) to ‘create a more unified and integrated organization that is linked to its strategy’ [10].This review placed greater focus at the beginning of the capability life cycle and the achievement of joint capability. It also established organizations within the Australian Defence Organisation (ADO) to undertake force design and joint capability integration to deliver more effective and integrated defence capabilities. As a consequence, there is a shift towards design and integration of capability above the level of individual systems and projects. This approach is being realized via a series of acknowledged capability programmes spanning the joint, air, land, maritime and intelligence domains. SoSE has the potential to underpin the development and execution of such capability programmes as well as the synthesis of joint warfighting capabilities across multiple programmes. The ADO will, therefore, need to develop and apply suitable SoSE methodologies to support this capability approach.

This paper starts with an introduction on capability and discusses how large-scale capabilities are realized through SoSE efforts that integrate multiple largely independent projects into capabilities. There is a widely held belief that it is necessary to design a specific SoSE methodology to meet each SoS as each has its own challenges [3, 22, 41, 55]. Consequently, the next section discusses an IDEF0 depiction of a methodology design and execution process showing its inputs, outputs, controls and mechanisms that provide context for use of the principles. The body of the paper then discusses the structure used to organize the principles along with the principles themselves.

15.2 Capability and Capability Engineering

The Australian Interim Capability Life Cycle Manual [9] defines capability as: ‘the power to achieve a desired operational effect in a nominated environment within a specified time and to sustain that effect for a designated period’. Capabilities are essentially the enterprise-level emergent properties of a system of interdependent elements referred to as the Fundamental Inputs to Capability (FIC) [9] that comprise personnel, organization, collective training, major systems, supplies, facilities and training areas, support and command and management. This paper is primarily interested in capabilities at the higher levels of complexity that span multiple military services and comprise major systems acquired and maintained via numerous systems engineering efforts across multiple organizations. Capability systems engineering [33] captures the type of SoSE that we are concerned with; it encompasses the design of the capability and also the planning and governance of the delivery of capabilities such as the UK air defence system.

Similarly, SoSE draws upon traditional systems engineering practices and concepts that are tailored to reflect the SoS context. However, there is usually no single owner of a SoS, and independent, concurrent management and funding at both the constituent system (CS) level and at the SoS level is the norm [2, 7, 35, 36, 39, 40, 41, 56]. Therefore, in order to achieve effective capability outcomes, it becomes necessary for key players to influence other parties (particularly independent project offices) on the basis of perspective, breadth of knowledge and analysis rather than from a position of authority [42]. It is primarily the lack of management control that makes SoSE fundamentally different from traditional project-based SE.

15.3 The Design of SoS Engineering Methodologies

This paper was motivated by the desire to provide some structure and traceability in the process of designing SoSE methodologies. It is, therefore, useful to clarify what we mean by a methodology. Jackson [27] states that a methodology is a kind of transferable problem-solving capability. He argues that a methodology facilitates, organizes and reflects on the use of methods, procedures, models, tools and techniques. Methodology establishes the principles behind the use of system models, architectural models and mathematical models. It draws on an agreed framework of ideas and operates on an agreed area of concern (class of problems). In this paper, the term includes not only the set of processes, tools, methods and techniques and their application but also the worldviews and philosophical positions that identify the value the methodology is seeking to create.

It is also useful to define the term *process* as used in this paper. A process is a set of activities that are interrelated or that interact with one another [24]. Processes have defined inputs and outputs and describe what needs to be done while leaving the selection of resources, methods, tools and techniques to the process user. Executing processes can be a routine matter or can be a highly intellectual activity such as undertaking a research process or a design process. An effective process produces the defined set of outputs to the quality level expected. Processes are one of the key elements of a mature methodology such as the systematic design process [43]. It should be noted that methodologies and their processes should not be thought of as being prescriptive but rather problem-solving approaches to be tailored and instantiated to suit the specific challenge of the SoS of interest.

Tailoring of SoSE approaches is a design activity and research on this for the ADO is an ongoing research activity in the Defence Science and Technology Group [13]. The two most important inputs to this are knowledge of the class of the SoS of interest and a set of principles, derived from practice and theory that can direct key design choices. The SoSE methodology design process can be modelled using the IDEF0 modelling language (Fig. 15.1) that explicitly records the process name, its inputs, outputs, mechanisms and controls.

The inputs are the SoSE problem of interest (capability challenge) and salient knowledge of the problem context including all elements that impact on SoSE

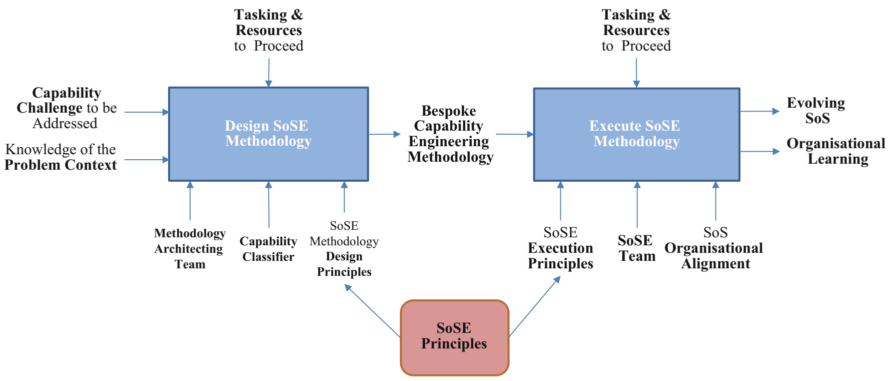

Tasking & Resources to Proceed
Tasking & Resources to Proceed
Capability Challenge to be Addressed
Knowledge of the Problem Context
Design SoSE Methodology
Bespoke Capability Engineering Methodology
Execute SoSE Methodology
Evolving SoS
Organisational Learning
Methodology Architecting Team
Capability Classifier
SoSE Methodology Design Principles
SoSE Execution Principles
SoSE Team
SoS Organisational Alignment
SoSE Principles

Fig. 15.1 An IDEF0 representation of the design capability integration methodology function

execution. The principle mechanism for undertaking the design task is a small team of methodology architects along with two supporting tools: a tool to classify the SoSE problem of interest [57] and the set of design principles under discussion in this paper. The control that initiates this process is a tasking order that supplies the resources to proceed. The IDEF0 diagram makes the control input explicit. Finally, the output is the bespoke capability integration methodology described at whatever level of resolution is needed to suit the task in hand. In common with all IDEF0 diagrams, this can be decomposed to provide visibility of the component design processes used to create the output and this will be the subject of subsequent papers. Figure 15.1 also shows that the SoSE principles subsequently become mechanisms to enable the execution of the methodology.

15.4 SoSE Methodology Design Principles

As stated above, a key enabler of the design process for a SoSE methodology is a set of principles that inform the design and use of the approach for a given challenge. This information is particularly necessary in the Australian Defence context due to a dearth of first-hand experience within the country on structured SoSE approaches.

Cook et al. [11] provided the first comprehensive literature review on the subject and identified 48 success factors for SoSE. Within that paper, the success factors were grouped by SoSE capability FIC elements that reinforced that all the elements are needed to perform effective SoSE practice. Also, as expected after one iteration, there were more principles to be found and subsequent reviews swelled the list to beyond 100 entries. However, it was found that the sheer number of initial principles made them difficult to operationalize and utilize. In addition, the principles varied in nature, profundity and level of abstraction, and as such a new taxonomy was needed.

It is useful to describe what we mean by the term *principles*. Relevant dictionary definitions of *principle* include: ‘a fundamental truth or proposition that serves as the foundation for a system of belief or behaviour or for a chain of reasoning’; ‘a general scientific theorem or law that has numerous special applications across a wide field’ (*www.oxforddictionaries.com*); ‘an adopted rule or method for application in action’; and ‘a fundamental, primary, or general law or truth from which others are derived’ (*www.dictionary.com*). These definitions work well when we are referring to the key principles that underpin SoSE practice and imply that principles cover everything from the fundamental tenets of a discipline through to empirical rules of thumb to guide action.

It would appear that many of the collected success factors or principles can be described as *heuristics*: defined from the engineer’s perspective as ‘statements of common, or contextual, sense that aid in concept development, problem solving, decision making or judgements’ [46, p18–19]. Rechtin states that heuristics ‘… differ from scientific laws by being more qualitative, more suggestive, and usually less amenable to replicable measurement …, they are generalizations from specific examples, not conclusions derivable from general principles’ [46]. He classifies heuristics into two classes: descriptive heuristics that relate to descriptions of the problem situation and prescriptive heuristics that relate to what to do about it.

Another influence on the structuring of the principles was the study by Adams [1] who proffered a structure for contributions to knowledge that ranged from philosophical through theoretical and methodological to techniques. Some of the principles presented here reside in the philosophical category in that they articulate a system of beliefs that provide a grounding for theories, whereas others are quite prescriptive on courses of action.

In light of these observations, the expanded list of principles was reviewed to seek themes and categorization options. This led to the development of a philosophical hierarchy structuring the principles. This three-tiered framework (Fig. 15.2) separates the philosophical underpinnings or belief system (worldview) from the conceptual constructs (concepts) and the principles necessary for their implementation (implementation principles). Based on this framework, a thematic analysis was conducted whereby each principle within the list was reviewed, assessed, categorized and further grouped. On completion of this process, it appears that the principles identified from the literature can be successfully placed within the framework. The result of these activities was the development and instantiation of a detailed and pragmatic framework to support the design and use of methodologies for SoSE. Each of these hierarchical layers is described below along with their content.

15.4.1 Tier 1: Worldview

The term worldview used here is equivalent to Checkland’s use of the word *weltanschauung*: ‘the (unquestioned) image or model of the world that makes this

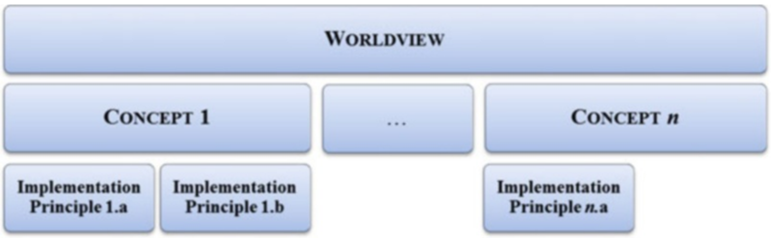

WORLDVIEW
CONCEPT 1
...
CONCEPT n
Implementation Principle 1.a
Implementation Principle 1.b
Implementation Principle n.a

Fig. 15.2 Philosophical hierarchy used in thematic analysis

particular human activity system (with its particular transformation process) a meaningful one to consider' [8]. The worldview encompasses the belief system upon which the concepts and implementation principles rest. Our overarching worldview statement is that in the Australian Defence environment, thoughtfully designed SoSE approaches will improve defence capability outcomes: both the capabilities delivered and their in-service effectiveness. The support for this comes from not only the burgeoning SoS literature but more importantly from anthologies of case studies in SoSE [22, 29, 30]. Within this overarching worldview also sit four additional pillars of belief that inform our concepts and implementation principles which are discussed in turn below.

SoSE Is Value Driven Defence capability outcomes are measured in terms of the stakeholder value over time that a delivered capability provides to the stakeholders (e.g. operators, owners and the public it is being used to serve) [49]. This element of the worldview is essential in keeping SoSE grounded and clarifies that the design and utilization of SoSE methodologies, just like SE methodologies, need to be targeted at maximizing stakeholder value (and hence return on investment).

SoSE Is a Socio-technical Activity Henshaw states 'many of the issues in SoS turn out to be non-technical; as such there needs to be a focus on the social, political, and enterprise aspects of SoS' [24]. In traditional project-centric engineering, project goals and lines of authority are usually apparent to all participants. The need for a constituent system to integrate into one or more SoS, a task for which it has often not been designed, blurs those goals and lines of authority, and the needs of the SoS are often in conflict with the project imperatives (a good example of this is given in the study by Stevens [50]). We believe that SoSE is inherently a socio-technical activity and to succeed substantial effort needs to be dedicated to the social, cultural, political and enterprise aspects of the SoS.

SoSE $\neq$ SE We believe that SoSE is fundamentally different from product-centric and project-centric SE. There are some that may contest this belief but it is hard to imagine that classical project-centric project management and systems engineering practices could ever have brought about a decentralized SoS such as the Internet. This principle is readily supported by the literature on defence SoSE methodologies [7, 25, 33, 36, 40, 41].

SoSE Is Multidisciplinary, Practice Based and Evidence Driven The main distinguishing characteristics of a SoS from a monolithic system are the managerial and operational independence of the constituent systems from which they are formed. Henshaw [24] points out that in order to cover the socio-technical dimension across independent project offices, multidisciplinary teams are essential to properly manage the emergent behaviours of the SoS. Furthermore, in common with all engineering endeavours, precedence needs to be given to methods, processes, tools and techniques that have been shown to work well in practice [41].

15.4.2 Tier 2: Concepts

The above statements form the guiding philosophy for the next level of categorization and development. The thematic analysis identified seven core groupings that aligned to the worldview, as shown in Fig. 15.3. A title and short description of each combination was then generated to represent the fundamental commonality observed. The concepts are largely descriptive heuristics, and it is proposed that the concepts described below can be utilized to provide direction and scope for the design and implementation of SoSE methodologies.

Enterprise Concept The enterprise must embrace SoSE [4, 5, 11, 20, 45, 51, 52, 57]. Enterprise-level actions must occur to support and facilitate the changes that SoSE requires. This includes education and training, responsibilities and roles and, more than anything else, cultural change.

Evolutionary Concept SoSE requires an incremental, evolutionary approach, one with long-term goal(s) and phased, implementable milestones that mark clear capability augmentation [25, 36, 38, 40, 41, 45, 46, 51]. The source articles stress the importance of the use of spiral-model deliveries within a broader long-term roadmap and the need to continuously determine and re-assess milestones based on pragmatic progress and changes in the context.

Methodology Concept SoSE methodologies must be tailored to the specific SoS, environments and missions [21, 23, 25, 26, 34, 38, 40, 41, 50, 55]. Blended approaches utilizing multiple methodologies and perspectives that can adapt over time as the SoS adapts are recommended. The complexity of the environment within which SoSE exists requires the understanding of multiple perspectives and multidisciplinary tools and techniques. Methodologies must be tailored to the circumstance and monitored and adapted to the continual change that is expected in a SoS.

Socio-technical Concept SoSE is socio-technical and human-based activity, actively combining organizational, personnel, infrastructure and technical aspects [4, 41, 45, 50]. SoSE requires the alignment of people, organizations, facilities and technology to meet enterprise goals. Trade and resource decisions must balance

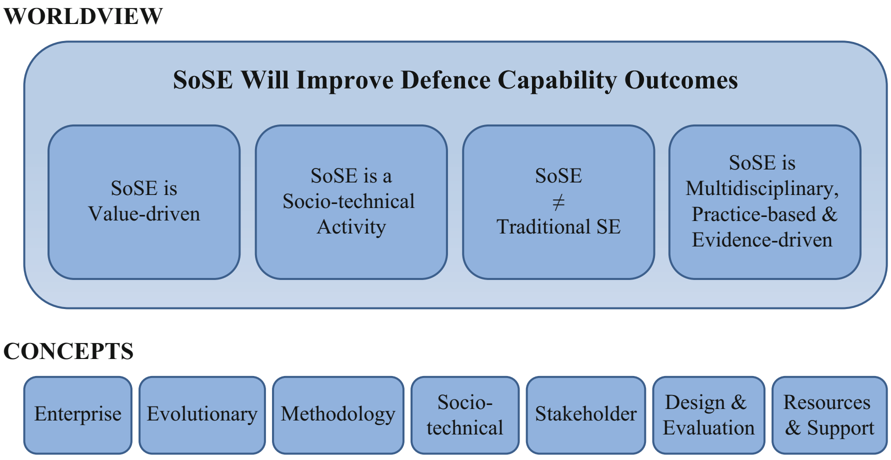


Fig. 15.3 The worldview pillars and the concepts layer

both technical and non-technical aspects, recognizing that key capability components must often be delivered by non-technical means.

Stakeholder Concept SoSE must make winners of key stakeholders from across the SoS and work within their values [3, 4, 6, 45].The sources place importance on the essential nature of stakeholder engagement, collaboration and their importance in the identification, development and delivery of SoS engineering and its outcomes.

Design and Evaluation Concept Blended top-down and bottom-up design and evaluation practices are needed to support evolution and increase overall system resilience; these must be light-touch, flexible and adaptive and supported by evidence-based assessments [3 ,4, 28, 40, 41, 47, 52]. This concept focuses on technical design as well as evaluation of the constituent systems and the SoS with emphasis on flexible and light-touch approaches to support SoS evolution. The ability of SoSE to both provide and receive direction is seen as essential to adaptivity.

Resources and Support Concept Resources (people, funding and facilities) and governance structures must be agile, collaborative, flexible and innovative [37, 45, 50, 51]. Appropriate resourcing is essential for the success of any activity; however, harnessing resources is a greater issue in SoS where they are usually distributed across constituent system (CS) project offices. Collaboration and innovation in resource utilization is key to ensuring SoS delivery.

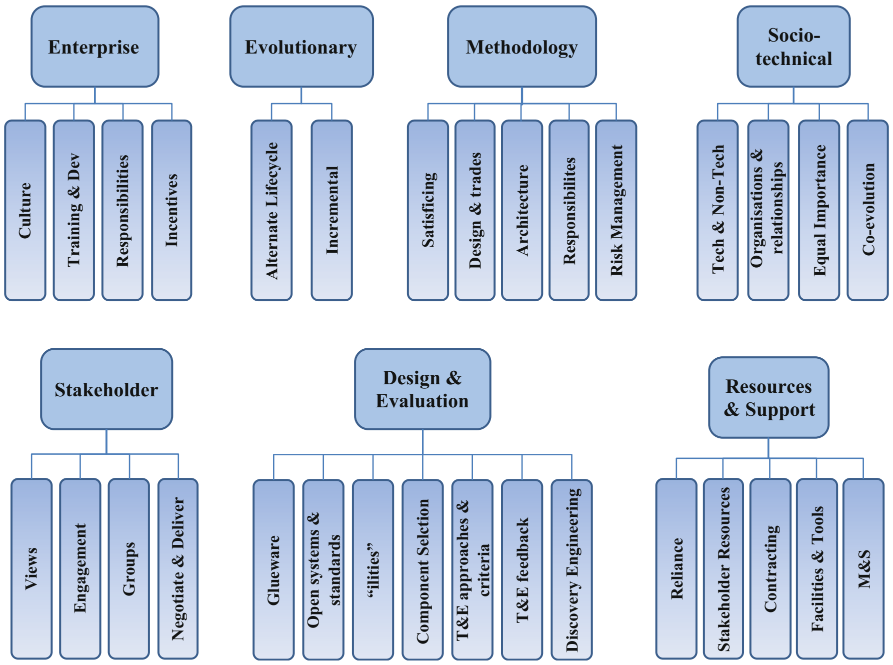


Fig. 15.4 Implementation principles

15.4.3 Tier 3: Implementation Principles

Implementation principles are those strategies, rules or guidelines directing conduct and action, which follow on directly from the concepts discussed above. The implementation principles are naturally grouped within the conceptual structure as a hierarchy and provide more detailed information on the practice and application of the concepts. This structuring supports a more pragmatic use of the principles. The implementation principles are presented below their overarching concept in Fig. 15.4 and are discussed individually below.

15.4.3.1 Enterprise Implementation Principles

The enterprise concept ('The enterprise must embrace SoSE') contains four primary implementation principles:

1. *Create and maintain a SoSE-aware culture*. Stakeholder organizations must intrinsically consider and balance the needs of both the SoS and CS. SoSE guidance should be understood and followed inherently by all those involved [4, 5, 57].

2. *Training, development and management of SoS engineers and stakeholders must be structured and specific for SoSE ($\neq$ SE).* Key competencies for all stakeholders must be identified and managed. This should be supported by appropriate education and training targeted towards SoSE [11, 20, 51].
3. *The enterprise must take on fundamental responsibilities and provide key services to facilitate SoSE.* Enterprises must take on responsibilities such as architecting and developing the SoS, setting and performing governance, sharing information and common tools and establishing research programmes to expand SoSE knowledge and capability [20, 45, 52].
4. *Incentives are necessary to reward and instil good SoSE behaviour.* To inspire appropriate behaviour in stakeholders (especially CS staff), the enterprise must incentivize for delivery of SoS outcomes not just CS outcomes. This requires the discovery of the attributes of good SoSE behaviour across the organization [17, 40, 41, 45].

15.4.3.2 Evolutionary Implementation Principles

The evolutionary concept, described earlier as 'an incremental, evolutionary approach;... with long-term goal(s) and phased, implementable milestones ...' comprises two primary implementation principles:

1. *SoSE should be incremental and evolutionary.* Initial SoSE should be investigatory and pragmatic, building and adapting the capability in stages. This principle establishes the need to incorporate new requirements, new technology and other innovations throughout the life of the SoS [6, 18, 25, 40, 41, 46, 51].
2. *SoSE utilizes alternate life cycle models.* Standard life cycle and development models are not appropriate for SoS capabilities. Fielding staged updates in pragmatic spirals allows the capability to be evaluated and then adapted to the environmental changes expected [3, 18, 40, 41, 45].

15.4.3.3 Methodology Implementation Principles

The methodology concept states that 'SoSE methodologies must be *tailored to the specific SoS*, environments and missions'. Five primary implementation principles were synthesized in this area:

1. *SoSE methodologies must be tailored to the specific SoS and seek satisficing not optimizing solutions.* SoSE does not seek to identify the best or optimal solution; rather it provides a 'good' or 'satisficing' solution. This places more importance on tailoring [21, 32, 41, 55].
2. *Design strategy and trades are a key focus for SoSE.* Throughout SoS evolution and particularly during initial SoS establishment, understanding and influencing the design methodology and trade spaces at both system and SoS level are key to successful SoSE [41].

3. *SoSE is informed (not driven) by a model-based reference architecture, but only as the capability matures*. Detailed top-down, architecture-driven approaches are resource intensive and are not well suited for the initial iterations of the SoS. SoS architecture descriptions have, however, been found to be effective in SoS coordination activities in later iterations [7, 38, 41, 45, 52–54].
4. *SoS owners are responsible for architecting and directing* the capability while allowing constituent *system owners to manage systems information*. Clear roles and responsibilities that support collaboration and efficiency will facilitate SoSE success. System owners provide confidence in systems information, while SoS owners deliver clarity in design and evaluation across SoS iterations [31, 41, 45].
5. *Use risk management to focus on key SoS aspects/outcomes and ensure balance in effort to achieve satisficing goals*. Risk management is used to maintain governance and drive decision-making in SoSE, particularly to direct satisficing for the whole SoS rather than optimizing for individual systems [6, 38].

15.4.3.4 Socio-technical Implementation Principles

Four key implementation principles were identified within the socio-technical concept ('*SoSE is socio-technical and human-based activity*, actively combining organizational, personnel, infrastructure and technical aspects'):

1. *Balance technical and non-technical aspects in SE trades and resource decisions*. Success depends on the ability of SoS managers to work across systems and balance technical and non-technical issues. This requires experienced, capable SoS managers and SE teams [41, 50].
2. *Understanding* the structures, policy and behaviours of the developer, implementer and user *organizations and their interrelationships is crucial* to SoSE delivery. Acknowledging and accounting for fundamental SoS stakeholder drivers, including policy, organizational structure and culture, and their interdependencies are essential to ensure that the SoS is accepted and milestones are achieved [45].
3. Delivery of *organizational, personnel and infrastructure elements is just as important as technical* delivery in a SoS capability. SoSE needs to undertake the design tasks holistically and coordinate delivery across all FIC elements [4, 41, 50].
4. *SoSE must co-evolve with all FIC elements to realize the potential of the SoS*. A typical SoS life cycle is of sufficient duration for other components, such as strategy, culture, organizations and doctrine, to evolve significantly. Therefore, SoSE must take their influence into account and reflect their impact [11].

15.4.3.5 Stakeholders Implementation Principles

The Stakeholders concept of 'SoSE must *make winners of key stakeholders* from across the SoS and work within their values' led to four primary implementation principles:

1. *Identify and understand each stakeholder's views: who is important and what success means/is for them.* This principle directs that it is not enough to simply identify all the stakeholders; it is also important to understand their perspectives, constraints, option/trade spaces, what they value and what value they deliver to the SoSE outcomes being sought [6, 45].
2. *Strong and positive stakeholder engagement must be developed and maintained throughout the life of the SoS.* Given the independent nature of constituent systems that form the SoS, and their asynchronous life cycles, continuous engagement of all stakeholders is a high priority in SoSE. Close relationships between all stakeholders improve the ability to shape thinking and achieve compromise [45].
3. *Treat stakeholders together as groups based on worldviews, roles, responsibilities and interests.* Grouping stakeholders assists in stakeholder community development, clarifying needs and focussing decision-making (but not for trade-offs – see next) [45].
4. *Negotiate the standard set of capabilities* and plans with key stakeholders. Identify and *rapidly deliver their most valued capability*. This principle encourages the stakeholders themselves to bargain for their desired outcomes, improves total SoS understanding and maintains commitment by delivering their prized capability [6, 45].

15.4.3.6 Design and Evaluation Implementation Principles

Seven primary implementation principles were attributed to the design and evaluation concept described as 'blended top-down and bottom-up design and evaluation practices . . . must be light-touch, flexible and adaptive':

1. *There is a need for 'glueware'.* SoSE benefits from the use of 'glueware' or bottom-up approaches to integration, particularly in early iterations and if the SoS is composed of very independent CS [40, 41].
2. *Open systems concepts and open standards must be used* to facilitate the interoperability needed to achieve emergence, adaptability and flexibility. The use of open standards, open systems techniques, loose coupling and modularity are crucial to support reconfigurability and interoperability [3, 40, 41, 52].
3. *Adopt a 'design for SoS 'ilities' approach; this will run in parallel with the SoS and constituent system (CS) design activities.* SoSE must design for non-functional as well as functional requirements. Non-functional characteristics include robustness, resilience, redundancy and interoperability [4, 28, 45, 47].

4. *Selection of implementation components (at CS level) should be guided by SoS and enterprise needs*. CS seek to optimize their own systems often at the penalty of broader integration and SoS/enterprise requirements. Guidance regarding these needs must be provided and rewarded to engage CS [52].
5. Appropriate and evolutionary SoS *test and evaluation (T&E) activities, approaches and success criteria must be tailored to the SoS operational need*. SoS T&E is fundamentally different from systems T&E. Evaluation and certification activities, processes and acceptance criteria for SoS must be driven by the operational context, objectives, constraints and risks. These must be re-assessed for each spiral. Systems testing must support and build towards integrated capability testing [4, 16, 57].
6. *Utilize test and evaluation (T&E) feedback in development and evolution of the SoS*. T&E is intrinsic to SoS life cycle feedback. It must be used to drive the next phase of SoS evolution. The implications of systems T&E cycles must be used in SoS evolution, T&E and planning in a Kaizen approach [12, 19].
7. *Use discovery engineering for continual SoS improvement and to support innovation and de-risk capability delivery*. Facilities to support experimentation and prototyping are critical to discover SoS capability issues, de-risk integration and extend perspectives. These provide platforms for CS to examine their contribution to SoS emergence and opportunities for innovation and collaboration [11, 31, 40].

15.4.3.7 Resources and Support Implementation Principles

The resources and support concept requires *resources* (people, funding and facilities) and governance structures *to be agile, collaborative, flexible and innovative* that delivered five primary implementation principles:

1. The SoSE team must *achieve much of its mission through the CS project offices*. The SoSE team is typically small [37] and relies on information and services from other organizational elements, primarily the CS project offices. Project office buy-in and consensus-based co-ordination is essential [41].
2. Achieve SoSE programme robustness through *securing resource support across the stakeholder network*. Marshalling resources from multiple sources within the stakeholder community encourages commitment, shares the SoS burden and reduces risk while creating resilience for the SoSE programme [45].
3. *Flexible and innovative contracting mechanisms* are required to ensure successful SoSE. Contracting arrangements for CS and SoSE staff must align with the SoSE methodology, context and SoS evolutions. This includes formal and/or informal agreements between CS and SoS engineering teams [11, 15, 38].
4. SoSE *facilities, infrastructure and tools must be more collaborative, federated and interoperable* and aligned to the methodology. Data consistency and knowledge management are essential. Facilities such as design and decision support environments facilitate communication between stakeholders. System

engineering tools to support SoSE are necessary to support and record design rationale [20, 40, 41, 50].

5. *SoS-focussed modelling and simulation (M&S) is essential for analysis and assessment*. Application of M&S is critical to support planning, trade decisions and evaluation throughout life cycle spirals [4, 48, 50].

15.5 Conclusion

This paper has provided an outline of two of the key aspects required to enable the development of SoSE methodologies to support ADO capability integration. The first aspect is an analysis of the salient ADO constraints and environment that provides the direction for tailoring SoSE methodologies to the context. The second is a philosophical framework that captures the worldview, concepts and implementation principles needed to perform methodology design. The source principles (from which these concepts and implementation principles were derived) were extracted from numerous sources spanning both defence and civilian application areas and worldwide research. While both the process for deriving the framework and the populated framework itself have been derived for a specific purpose in mind, they are applicable to the development of any SoSE methodology. Further work will elaborate the detail of the process shown in Fig. 15.1, identify how the principles would be used by different stakeholder groups, elaborate the required mechanisms and evaluate the framework through case studies.

References

1. Adams KM (2011) Systems principles: foundation for the SoSE methodology. Int J Sys Sys Eng 2(2-3):120–155
2. Allison JS, Cook SC (1998) The new era in military systems thinking and practice. Proceedings of Systems Engineering 1998. Anteroinen, J. 2012. The holistic military capability life cycle model. 7th International Conference on System of Systems Engineering (SoSE), 16–19 July 2012, pp 167–172
3. Azani C (2009) An open systems approach to system of systems engineering. In: Jamsidi M (ed) System of Systems engineering: innovations for the 21st century. Wiley, Hoboken, pp 21–43
4. Barot V, Henson S, Henshaw M, Siemieniuch C, Sinclair M, Lim SL, Jamshidi M, DeLaurentis D (2012) SoA Report: State of the Art on Systems of Systems Management and Engineering. Final Report of the Trans-Atlantic Research and Education Agenda in Systems of Systems (T-AREA-SoS), Work Package 2, Deliverable D2.1, Version 2. No. TAREA-PU-WP2-D-LU-9
5. Blockley DI, Godfrey P (2000) Doing it differently: systems for rethinking construction. Thomas Telford
6. Boehm B, Lane J, Koolmanojwong S, Turner R (2014) The incremental commitment spiral model: principles and practices for successful systems and software. Addison-Wesley. ISBN-13: 978-0-321-80822-6

7. Camm DA, Fitchett AM (2011) Solving complexity challenges in the capability lifecycle – the MOD System of Systems approach. INCOSE UK Conference, 2011
8. Checkland P (1981) Systems theory, systems practice. Wiley. ISBN 0 471 27911 0
9. Commonwealth of Australia (CoA) (2016) Interim capability lifecycle manual. Department of Defence, ACT
10. Commonwealth of Australia (CoA). 2015. First principles review: creating one defence. Australian Department of Defence. http://www.defence.gov.au/publications/reviews/firstprinciples/Docs/FirstPrinciplesReviewB.pdf
11. Cook SC, Nowakowski S, Unewisse M (2013a) Principles for designing system of systems engineering approaches for the Australian defence force. Proceedings of SETE 2013, Canberra. ISBN 978-0-9752028-7-6
12. Cook SC, Nowakowski S, Unewisse M (2013b) Towards an SoS engineering approach for integrating Australian defence force capabilities. Proceedings of SETE 2013, Canberra. ISBN 978-0-9752028-7-6
13. Cook SC, Pratt JM (2014) Towards designing innovative SoSE approaches for the Australian defence force. Proceedings of IEEE 2014 International Conference on System of Systems Engineering, Adelaide, Australia, June 2014, ISBN: 978-1-4799-5227-4
14. Cook SC, Pratt JM (2016) Typology dimensions for classifying SoSE problem spaces. 11th IEEE System of Systems Engineering Conference (SoSE), Kongsberg, Norway, pp 1–6
15. Cook SC, Unewisse M (2011) A survey of defence industry systems engineering and systems integration capability: part 2: qualitative results and survey findings. Proceedings of SETE 2011, May 2011
16. Dahmann J, Lane JA, Rebovich G, Lowry R (2010) Systems of systems test and evaluation challenges. In: System of Systems Engineering (SoSE), 2010 5th International Conference on, pp 1–6. IEEE
17. Dahmann J, Baldwin K (2011) Implications of systems of systems on system design and engineering. In: System of Systems Engineering (SoSE), 2011 6th International Conference on, pp 131–136. IEEE
18. Dahmann J, Rebovich G, Lowry R, Lane J, Baldwin K (2011) An implementers' view of systems engineering for systems of systems. In: Systems Conference (SysCon), 2011 I.E. International, pp 212–217. IEEE
19. Dahmann J (2012) Integrating systems engineering and test & evaluation in system of systems development. In: Systems Conference (SysCon), 2011 I.E. International Systems Conference, pp 1–7. IEEE
20. Freeman GR (2012) Workplace of tomorrow. Conference on Systems Engineering Research
21. Gorod A, Sauser B, Boardman J (2008) System of systems engineering management: a review of modern history and a path forward. IEEE Syst J 2(4)
22. Gorod A, White B Ireland V, Gandhi SJ, Sauser B (2014) Case studies in system of system, enterprise systems, and complex systems. Taylor and Francis Press. ISBN 978-1-4665-0239-0
23. Grisogono AM, Ryan A (2003) Designing complex adaptive systems for defence. Proceedings of SETE 2003
24. Henshaw M (2015) Good practice in systems of systems engineering (SoSE). In: SCI276 lecture series. CSO, NATO
25. ISO 9000 (2005) Quality management systems – fundamental and vocabulary. International Standards Organization
26. Ireland V (2011) Problems in treating a SoS as a PMBOK or SE style project. Introduction to System of Systems Engineering – Course Notes, DSIC
27. Jackson MC (2000) Systems approaches to management. Springer Science & Business Media, Boston
28. Jackson S, Ferris T (2013) Resilience principles for engineered systems. Syst Eng 16 (2):152–164
29. Jamshidi M (2009a) Systems of systems engineering: principles and applications. Wiley, Hoboken. ISBN: 9781420065886

30. Jamshidi M (2009b) System of Systems engineering: innovations for the 21st century. Wiley, Hoboken. ISBN: 9780470403501
31. Kalawsky RS, O'Brien J, Chong S, Wong C, Jia H, Pan H, Moore PR (2013) Bridging the gaps in a model-based system engineering workflow by encompassing hardware-in-the-loop simulation. Sys J IEEE 7(4):593–605
32. Keating C, Rogers R, Unal R, Dryer D, Sousa-Poza A, Safford R, Peterson W, Rabadi G (2003) System of Systems engineering. Eng Manag J 15(3):36–45
33. Kemp D, Daw A (2014) Capability systems engineering guide. INCOSE UK
34. Kline SJ (1995) Conceptual foundations for multidisciplinary thinking. Stanford University Press, Stanford
35. Kwon YS, Cook SC (2010) Considerations on the application of capabilities based planning to the SoS environment. APCOSE 2010, Keelung
36. Lalancette C, Lizotte M, Nécaille C, Robbins W, Waruszynski B (2008) Capability engineering within Canadian defence: experimentation and lessons learned. Proceedings of INCOSE 2008 Symposium. Utrecht
37. Lane JA, (2009) Impacts of system of system management strategies on system of system capability engineering effort. PhD thesis, University of Southern California
38. Luxeaux D, Ruault J-R, Wippler J-L (2011) Complex systems and systems of systems engineering. Wiley, New York
39. Maier MW (1998) Architecting principles for systems-of-systems. Syst Eng 1(4):267–284
40. Norman DO, Kuras ML (2006) Engineering complex systems. In: Complex engineered systems. Springer. ISBN 978-3-540-32831-5, pp 206–245
41. OSD (Office of the Secretary of Defense) (2008) Systems engineering guide for systems of systems, Version 1.0, Washington, DC: Office of the Deputy Under Secretary of Defense for Acquisition and Technology, Systems and Software Engineering, ODUSD(A&T) SSE
42. Oxenham D, Swales S (2010) Systems people: description of the four stages of systems skills, dstl. UK MoD
43. Pahl G, Beitz W, Feldhusen J, Grote KH (2007) Engineering design: a systematic approach. Springer, Berlin
44. Pratt JM, Cook SC, Unewisse MH (2015) A SoSE principles framework to drive Australian defence capability integration. Proceedings of SETE. Canberra. ISBN: 978-1-877040-74-0
45. Rebovich G Jr, DeRosa JK (2011) Patterns of success in systems engineering: acquisition of IT-intensive government systems., Report No. MP110369. MITRE CORP, Bedford
46. Rechtin E (1991) Systems architecting: creating and building complex systems. Prentice Hall, Englewood Cliffs
47. Ricci N, Fitzgerald ME, Ross AM, Rhodes DH (2014) Architecting systems of systems with Ilities: an overview of the SAI method. Proc Comput Sci 28:322–331
48. Rumford GJ (2008) Joint mission environment test capability JMETC. US DoD
49. Sillito H (2014) Architecting systems – concepts, principles and practice. College Publications, Londres. ISBN 978-1-84890-154-4
50. Stevens R (2011) Engineering mega-systems. CRC Press, Boca Raton. ISBN 978-1-4200-7666-0
51. Turner R, Verma D, Weitekamp MR, Telford A (2009) A distillation of lessons learned from complex System of Systems acquisitions. NDIA Systems Engineering Conference
52. UKMoD (2013). System of Systems approach (SOSA) open systems strategy. UK Ministry of Defence
53. USN (2006a) US Naval "Systems of Systems" Systems engineering guidebook Volume 1, ASN(RDA), US Navy, April
54. USN (2006b) US Naval "Systems of Systems" Systems engineering guidebook Volume 2, ASN(RDA), US Navy, April
55. USAF (2005) Report on system-of-systems engineering for air force capability development. SAB-TR-05-04, USAF Scientific Advisory Board, July 2005

56. Valerdi R, Axelband E, Baehren T, Boehm B, Dorenbos D, Jackson S, Madni A, Nadler J, Robitaille P, Settles S (2008) A research agenda for systems of systems architecting. Int J Sys Sys Eng 1(1/2):171–188
57. Wilson B, Wissink T, Dahmann J, Palmer J (2010) Test and evaluation issues for systems of systems: sleepless nights to Sominex. NDIA Systems Engineering Conference

Chapter 16
System Analysis and Verification: A Comprehensive Approach and Case Study

Haifeng Zhu, Mark Moulin, Brian Murray, Vladimir Fonoberov, and Igor Mezic

Abstract Large complex systems such as systems of systems are difficult to analyze. They are especially difficult to formally verify primarily due to the state-space explosion. This paper addresses this issue and presents a comprehensive approach on a case study of a public domain multiple-unmanned aerial vehicle planning and search algorithm. The studied model contains both high-level swarm logic and low-level non-deterministic/probabilistic communication routing and error details, and its complexity exceeds our formal verification tool's capability. Our process is generic and performed with two kinds of methods (formal method and traditional stochastic methods) together "co-picturing" the system's performance. We introduce algorithms to obtain more information about the system's behaviors, than either of these two methods applied alone. It utilizes off-the-shelf tools and minimizes additional programming development. Large amount of simulations and statistical analyses together with formal verification are performed to demonstrate the approach's feasibility, and useful experiences are shown through the included extensive case study.

Keywords Formal verification • Sensitivity analysis • Complex systems

Brian Murray performed this work when employed by United Technologies Research Center.

H. Zhu (✉) • M. Moulin
United Technologies, East Hartford, CT 06108, USA
e-mail: zhuhf@utrc.utc.com

B. Murray
Livonia, MI 48150, USA

V. Fonoberov
AIMdyn, Santa Barbara, CA 93101, USA

I. Mezic
AIMdyn, Santa Barbara, CA 93101, USA

University of California, Santa Barbara, CA 93105, USA

A.M. Madni et al. (eds.), *Disciplinary Convergence in Systems Engineering Research*, https://doi.org/10.1007/978-3-319-62217-0_16

16.1 Introduction

Modern aerospace systems under development rely much more heavily on software for functionality than in the past. Their control implementation is difficult and error-prone especially in complex and tightly integrated embedded systems, which are becoming larger, more distributed, more heterogeneous, and possibly not deterministic.

Autonomous UAV (Unmanned Aerial Vehicle) swarm [1] is an example of such a system of systems. Due to the uncertainties that UAVs are facing during flights, complete analysis to these systems is an important issue to study in automation science and systems engineering. This paper studies a complicated UAV swarm example case to explore a comprehensive approach for the analysis and formal verification of UAVs [2]. In addition to formal methods, we apply other methods as well. In this paper, we call all methods that are different with what generally are considered formal methods as non-formal methods. An earlier study reported by Zhu et al. [3] shows the preliminary results using a different approach that applies both non-formal and formal methods independently and also did not present the details of their algorithms. This paper presents the details of our methods and results on combining these two kinds of methods together to capture the performance.

Formal methods provide a full-coverage analysis of the system behavior by creating a mathematical model of the system and examining whether all possible states of the system satisfy particular properties or characteristics. It is useful, however, often has the well-known state-space explosion problem where the number of system states grows quickly as the system complexity increases. In order to increase the use of formal methods in industry, tackling the state-space problem and adopting lightweight approaches are important. Here, lightweight refers to that the approach that should be achievable in a simple industrial environment that does not have many sophisticated formal verification tools and should avoid significant amount of tool development or programming as much as possible.

The major contributions of this work include: First, here we attempt to study a more complex UAV algorithm than many of the prior UAV formal verification example cases we know about (surveyed below), particularly exceeding the capability of formal verification tool utilized. Second, a set of methods were developed and demonstrated to combine non-formal and formal methods to "co-picture" the performance related to the same parameter, and it provides more information than either the non-formal or formal method applied alone. Third, our methods are based on off-the-shelf commercial tools and can be carried out in a lightweight manner based on experience. Even though our work is devised to tackle a UAV example, the approach and methods are generic and are not limited to UAV swarm or stochastic systems.

The goal of the UAV swarm studied here is to search and identify a hostile target. The goal of our study is to find out the performance and properties associated with such a searching goal, such as time to find the target, etc. In these systems, the decentralized controller of each UAV makes path planning according to its and other UAVs' locations. These locations are transmitted in the form of messages passed between air vehicles. These controllers are not able to make an optimal online path planning in the absence of global system status data. Communication failures and message delays affect the decision and performance of the controllers. In fact, one of the popular designs of embedded systems is using time-triggered communications to propagate information inside a system and using asynchronous communications (such as Wi-Fi) to exchange information among different systems. Such asynchronous communications may increase system model's dynamics causing more difficulties for formal verification. In the following, we will first review some of the existing work on UAV performance study and verification, indicate their differences with our work, and then introduce our approach, after which further comparison with some relevant formal verification approaches is discussed.

16.1.1 Formal Verification on UAV Algorithms

In a multilayer robotic architecture [4], typically swarm or cooperative search algorithms are in the high levels, and control/communications are in the lower levels. The following discusses existing UAV formal verification published works in both these levels. Then, we will discuss the application of statistical and probabilistic formal verifications in this area.

In cooperative search, [4, 5] build Kripke structures of UAV swarm search and use the Symbolic Model Verifier to perform verification on their properties such as deadlocks. However, they assume that inter-UAV communications are instantaneous and noiseless and have unbounded communication range. This is generally not possible in reality and removes the part of the non-determinism or probabilistic behaviors from the modeling, but this part is generally the hard part of the problem in verifications. Saberi et al. [6] discuss verification of a multi-robot system but not specifically on UAVs. They assume an unbounded planar surface, and each robot starts from an initial position and tries to move toward its destination, instead of swarming to find a target. Maxa [7] focuses on the routing protocol but not the search algorithm. Konur et al. [8] study swarm of foraging robots instead of surveillance and do not model details in both high and low levels such as multi-robot communication and collaboration details like ours. Our example case makes the system complex and poses challenges to formal verification. Finally, Massink et al., [9] use a description language Bio-PEPA to model swarm robot transport problem, which is a different problem from surveillance. Bio-PEPA allows different kinds of analysis (simulation, statistical model checking, and fluid analysis) in one single framework; however, Massink et al. [9] did not show how they could co-picture the same metric.

There are also a lot of work (e.g. [10–12]) in the motion and task planning that use formal verification techniques, but deal with a different kind of problem, namely modeling the system and tasks with certain logics and performing synthesis. They typically do not deal with complex stochastic system's state-space explosion issues. For example, the study by Belta [13] discusses abstraction methods for robotic swarms but it focuses on logic and specifications, instead of directly dealing with state-space explosion issues for our model checking methods and tools. Chaki and Giampapa [11] propose an approach to compute quantitative utility of robotic missions with linear temporal logic formulas but did not model wireless communication details as we do [12, 14] model planning specification with certain logic to perform analysis; however they do not explicitly model many details of UAVs including complicated communications, and it is not clear if their specific frameworks are capable of doing so.

In addition, there are also some robotic studies (e.g. [11, 12]) using probabilistic formal specification or verification methods (e.g. [15]).They may also face the same problem of state-space explosion when complexity is high.

16.1.2 Our Work

In this paper, we attempt to select a UAV formal verification example [16, 17] that is more detailed and complex than the relevant prior UAV works surveyed earlier in this paper. As briefly mentioned earlier and described with more details here, we attempt to use model checking to verify UAV swarm properties and obtain performance evaluation on target capture ability, effectiveness of asynchronous communication protocol, UAV collisions, and live-lock situations. In terms of tools, in this effort, we are required to use Mathworks Matlab/Simulink for coding and verification, as it is available in many industrial environments. This makes our work much more realistic, because Matlab/Simulink is a popular tool used in industry for productions. The methods to be developed are required to deal with all possible issues from this tool setup.

As expected, the main challenge is that our tool takes a significant amount of time to perform verification on our UAV models such that we were not able to see its termination, which we call as "Excessive Time/Seemed Non-Termination" (ESTNT) issue. The inherent state space explosion limits this method's capability from verifying such a large model efficiently. This problem is quite common. Even with potentially more powerful tools, when systems are more complex, in general, the tools may still have the same problem.

On the other hand, other approaches may be used for the analysis of asynchronous systems with parametric uncertainties. Statistical analyses or simulations provide a good picture of input/output relations, main system dynamics, and frequency responses. Therefore, we introduce a comprehensive approach that includes both types of methods in attempt to analyze system behavior and correctness as much as possible, especially when formal verification can only provide limited results due to high complexity. The steps in our approach are:

1. Perform sensitivity analysis. The analysis is performed on the full model (Model 1 that includes 3 to 13 UAVs) which is a partial implementation and abstraction

of a published swarm algorithm in the public domain [16, 17]. A sensitivity and uncertainty analysis tool, GoSUM [18], is used on Model 1 to determine the system dynamical behaviors (e.g., distribution of mean search time). Even though, the formal verification tool may take a long time without returning any results, GoSUM is always able to terminate in a controllable manner and provide a probabilistic observation of the system behavior. This is the first means to battle the ESTNT issue.

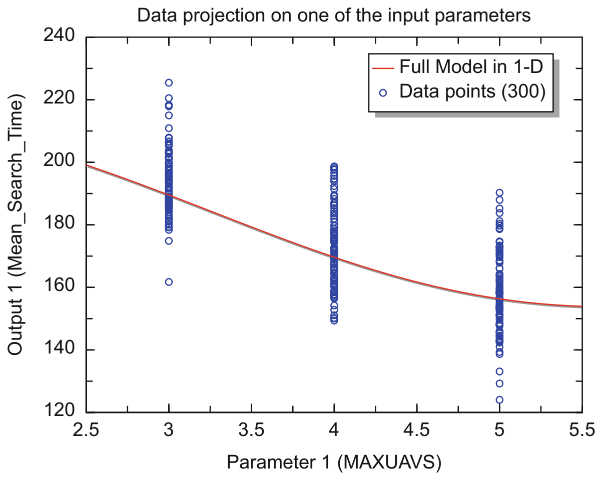
2. Obtain reduced models. For each data point result (e.g., at a specific number of UAVs) in Step 1, a reduced model is created and further simplified with approximations (thus becomes Model 2). This allows the UAV numbers to be fixed at different numbers, reducing model complexity to enable formal verification. The reduction and approximation, if any, should be performed without affecting the property to be analyzed. This is the second means to tackle complexity.
3. Perform formal verification on the reduced models. Model 2 is formally verified with Design Verifier to obtain additional results that are exact and cannot be obtained in Step 1. Formal analysis can be performed to understand more about the system behaviors, including parameter bounds and configuration tuning. These results can complement and enhance the results in the same performance picture (e.g., Fig. 16.2a) already obtained in Step 1 (i.e., co-picture process). For example, a lower bound of the search time can be obtained, if one wants to decide if a search should be given up when only limited fuel is available. Configuration tuning provides information to humans who attempt to determine the best deployment. The methods are generic and can be applied to other similar parameters than search time.

Together with the algorithms developed inside, this process is lightweight and captures both dynamical and static behaviors of the system under study and is able to provide a more complete picture than any of the methods (either non-formal or formal) applied alone. We will show the process, algorithms, modeling techniques, and results through simulation, analysis, and formal verification, as a multidimensional comprehensive approach to tackle large systems.

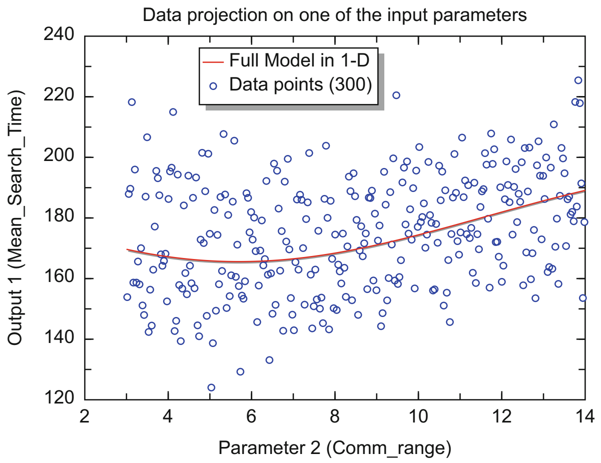
Our approach is different from simulation/sampling-based statistical verifications [19, 20, 21] that can potentially lead to errors because of the nature of limited simulations. Although postsimulation analysis such as hypothesis testing can be done easily within our Step 1 optionally as well, our Step 3 (formal verification) can obtain exact results (e.g., bounds) that are error free than simulation-based statistical verification, and we put results from Steps 1 and 3 together to co-picture the performance.

In comparing with the study by Clarke et al. [22], only our Steps 2 and 3 make use of approximations and counterexamples. The study by Clarke et al. [22] does not have the big picture stochastic analysis as in our Step 1. For complicated systems, Clarke et al. [22] may arrive in abstractions that are close to the original system model; thus, the verification may have the ESTNT issue, while our Step 1 (stochastic analysis) will always terminate, as its iterations are controlled. It may

Fig. 16.2 GoSUM analysis of the mean search time



(a)



(b)

not converge within a short period of time for very large systems, but in many cases, it does return useful observations, rather than no result at all after a long run time. [23] has similar differences as those with the study by Clarke et al. [22], except that it can provide stochastic results in our Step 1 when the state space is within its handling capability, in which case it can provide more accurate results than our approach. Otherwise, when the complexity exceeds the tool's capability, which is the situation we are dealing with, it may not return any results.

Therefore, our work is different from statistical model checking and basic abstraction refinement type of approaches. In addition, these relevant approaches are not supported by Design Verifier when this work was performed, thus not applicable, as our lightweight goal ideally should be no programming or tool development.

The remaining sections of this paper are organized as follows: Section 16.2 describes the example case under study, Model 1, and the sensitivity analysis with GoSUM; Section 16.3 performs formal verification on Model 2; and Sect. 16.4 concludes the study.

16.2 Example Case under Study

In this section, we present the details of the UAV example studied here. This example includes a high-level algorithm that performs target search and a low-level communication protocol that has probabilistic behavior. We use a similar approach as used in the study by Gaudiano et al. [1] to adopt a swarm strategy (similar to the study by Pack and Mullins [17]) on top of the UAV communication facilities (similar to the studies by Ladowski [16] and Karp [24]) to form the structure of a multilevel architecture UAV. The performance of search is impacted by the communications among different UAVs and relies on the knowledge of other UAVs' locations. A failure in the communication leads to incomplete status update on the other UAVs, which causes a non-optimal decision in the search procedure. A set of such UAVs was implemented by us in Matlab for a two-dimensional search area, as Model 1.

16.2.1 UAV Search Algorithms

The search task consists of choosing a path for each UAV such that a target is found. It is under the assumptions that all UAVs have the same motion capabilities and are given enough time. The search algorithm determines the UAVs' next positions in the search space and what information must be transmitted among the UAVs. Our implementation works according to the following rules [17]:

1. Given the current cell position of the UAV, identify the closest unvisited cell as the next cell to visit.
2. A UAV can move to the left, right, up, and down along the cells; movement to a diagonal cell is forbidden.
3. When multiple unvisited cells are equally close to the current cell, select the farthest cell from the location of other UAVs.
4. When multiple cells are equally far from the other UAVs, pick the cell that lies along the same direction the UAV just moved toward.
5. If multiple cells satisfy all the above conditions, an arbitrary cell is picked.

Additional assumptions used in Model 1 mainly include those regarding communications, for example, the search algorithm for each UAV uses constant updates of other UAVs locations, and communications contain both synchronous and asynchronous modes.

Our experience through simulations shows one may be able to reduce the number of UAV visits to the same cell through proper initial placement of the UAVs. For example, an arbitrary selection of initial placement can cause UAVs to cross each other's paths and visit the same cells more while leaving some common corner neighborhood uninspected for a period of time. As another example, when placing the UAVs too far apart, as the planning decision is made sometimes with incomplete global information due to communication failures, a UAV may choose a cell previously visited by other UAVs.

16.2.2 Communication Protocols

UAVs use wireless communication to coordinate swarm activities. To simulate UAV communications, Ladowski [16] chooses IEEE 802.11 ad hoc mode where each UAV acts as both host and router. The routing capability of each UAV permits transmission of data through neighboring UAVs. This increases the effective communication range of the entire network without increasing the physical transmission range of any particular UAV [24].

Whenever a UAV moves from one cell to another, it broadcasts its new location to all other UAVs. The communication then follows the routing policy and tries to find a path to reach those UAVs that are not in range. In this paper, UAVs use GPSR (Greedy Perimeter Stateless Routing) as a location-based routing protocol with the greedy forwarding strategy [24]. GPSR is a geographical information-based protocol that makes use of absolute positioning service.

A UAV uses GPSR to search all its neighbors in the communication range, finds the one that has the shortest distance to the destination, and sends the packets to it. This UAV neighbor that receives the packets will forward them in the same way, until the packets reach the destination UAV. Still, this forwarding can fail. One of the failure cases would be that no UAV exists in the neighborhood within the communication range of receiving UAV. In such a case, the search algorithm of the UAVs that have not received the updates would use the old location information, resulting in potential problems for searching logic. Some of these failure scenarios create an uncertainty factor in the overall search mission. Model 1 implements these algorithms, together with a simplified communication model where the failed messages can be transmitted again in asynchronous mode. Figure 16.1 shows this model where each data transfer contains possibly a second chance TX-2 in the asynchronous section. Model 1 is the standard reference model in our study.

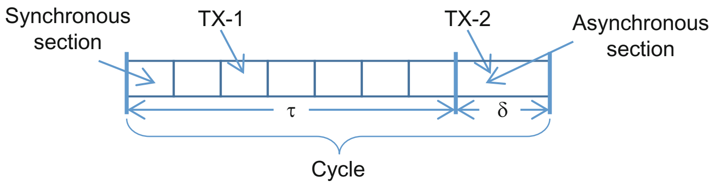


Fig. 16.1 Asynchronous communication model

16.2.3 Sensitivity Analysis

We implemented Model 1 with Matlab functions and it contains a 20 × 20 grid search space of 400 equal-sized cells that contains 1 target. The UAV speed is one cell per time unit, and the setup contains configurable number (sometimes up to 13) of UAVs. The search is considered complete when the target is found. With Design Verifier, formally verifying Model 1 takes excessively long time that we were not able to see its termination. Therefore, as the first step of our approach, GoSUM analysis is performed on Model 1 with random target and UAV positions, to obtain the true underlying dependence of the output on model parameters. The analysis of the swarm model shows that the mean mission search time (until target is found) depends approximately linearly on the number of UAVs and depends non-linearly on the communication range parameter (Fig. 16.2). The result was based on 300 samples each with 100 realizations. The analysis provides sensitivities with respect to all model parameters. Figure 16.2a shows heavy statistical masses following a declining slope, which is reasonable, as the more UAVs are used, the lower the mean search completion time tends to be. This provides a big picture of the whole system performance.

Through simulations and GoSUM analysis on our setup, communications are found to have other impact to the search behaviors. When communication is weak (i.e., small range, high error rate, etc.), UAVs lose updates from other UAVs and rely more on autonomous decisions instead of cooperative search. Therefore, our setup is roughly divided into two system operational modes based on the communication range parameter. We call a mode as *cooperative search* when the communication range is greater than six units in our specific experimental setup. In this case, on many occasions, each UAV tracks other UAVs' locations effectively, and all UAVs operate cooperatively. *Autonomous search* mode is called when communication range is less than six units, where on many occasions, each UAV has poor updates about other UAVs' locations and thus the UAVs operate autonomously at least during some of the time. In short, the above analyses can quickly produce relationships among different parameters that move our attention to the important ones and devising methods to verify them. Next, we will proceed to Steps 2 and 3 of our approach.

16.3 Model Reduction and Verification

Even though it is not possible to use Design Verifier to formally verify Model 1, it is still possible to obtain more results to further enhance Fig. 16.2 and reveal more system behaviors.

Note that the main issue with limited runs with GoSUM is that the results it provided may not be exhaustive. Therefore, the actual mean search time range may be larger than what is shown in Fig. 16.2. In this section, specifically, we use the case of three UAVs to illustrate our approach. GoSUM is not able to provide an accurate search time lower bound with limited runs, and it is not clear how many runs are needed to provide a real lower bound.

Having said this, stochastic analysis in Step 1 does provide a big picture of the whole system behaviors, which is a useful result already. From Fig. 16.2a, it can be expected that with quite some chances, the search time of more UAVs will be less than those with three UAVs.

The study on Model 2 continues to contribute more performance data to enrich our understanding of performance shown in Fig. 16.2a. Besides lower bounds, other performance measures may be studied as well. This is like zooming into a local part of the performance figure to understand more. In fact, a set of system properties were checked in our study to determine, for example, if the initial placements of the UAVs and the target affect the mission success, if asynchronous communication decreases the search time, or if the system encounters any collision or live-lock situations. Specifically, here, we summarize the study on target capture ability that is relevant to the search time that is our main interest. We define property F as "For a given initial placement of the UAVs, a target, and a set of communication parameters, a target is found (captured)" in less than a specified time unit that is set by the bounded model checking. Note that for our system, due to stochastic behaviors, F may only be true for part of the runs. However, we will show how we can still use it to obtain more system behavior information.

16.3.1 Model Reduction

As mentioned earlier, formal verification faces the challenges from the state-space explosion. Although many improvement techniques have been proposed during the past, practical tools still have difficulties in handling large systems. With only three full models of UAVs (Model 1) that include all the details of search collaborations, communication routing and failures, etc., our tool is still not able to terminate verification for a long time. Therefore, further reduction is needed, as described below:

1. Structures that are implemented in an overly complicated way are replaced with a simpler implementation. Structures that are not well supported by Design Verifier are replaced with those supported, which was already done in Model

1 but need to be done again, as some implementation simplification conducted may accidently use some structures that are not supported.
2. Functional blocks that are not relevant to properties to be verified are removed and replaced with proper structures that supply minimally needed values or sometimes simply terminators under the conditions that they would not change the code functionality such that properties to be checked are affected. For example, a display or data collection module that is not affecting the search time can be removed. Additional simplifications were applied to cut blocks, variables, and their allowed values, and these are judged by domain knowledge to ensure the search time is not increased. This additional simplification can affect the design, which is why domain knowledge is required; however, it can reduce the complexity more. It is effective, through our experience in this case study, but should be conducted by domain experts as less as possible as long as the model is simplified enough to terminate formal verification in a reasonable time that users can tolerate, in order to reduce chances of introducing more gaps or errors in the parameter being studied. We used this method to merge several communication failure modules together and, thus, cut the complexity down.
3. Data types are optimized to the smallest data types that are necessary. For example, if a variable only has two values 0 and 1 but is coded with integer type, it will be re-coded as Boolean type.

The resulted model (Model 2) has the following reconfigurable control parameters, which is called a *configuration*:

- The number of UAVs in the swarm
- Communication range
- Communication failure rate
- Asynchronous communication availability rate
- Initial positions of UAVs and the target

Although only three UAVs are demonstrated, other values than 3 can also be studied repeating the same method. However, as Fig. 16.2a shows, it may be conjectured that the mean search time would have quite some chances to be lower. Note that as this was found by GoSUM with limited runs, the result is statistical in nature and may not always be the case depending on the configurations. Further improving the accuracy is a different topic and may be studied in the future. Here, we target on showing the whole process of analyzing complex systems; therefore, the case study just uses only one example to show the process.

16.3.2 Formal Verification Methodology

In this section, we apply formal verification on the Model 2 with bounded model checking, where two techniques are developed: Algorithm 1 is to find a lower bound of the search time under a specific configuration, and Algorithm 2 can potentially provide human information for configuration tuning and in some cases obtain a feasible configuration. Both algorithms can be completed manually using Design Verifier in our case study, although in a more generic case, one may need to add small amount of programming to complete the tasks more efficiently.

Let us denote Nc as the time step bound utilized during verification. The following algorithm determines within Nc time steps, if the UAVs are impossible to find the target, under a specific configuration (e.g., the configuration used in Fig. 16.2a at UAV number 3).

Algorithm 1

1. Specify system property F "with negation" and verify.
2. If it holds, Nc is a lower bound of the search time for this configuration.
3. Otherwise, it means it is possible that UAVs can find the target within Nc steps, and one just changes another smaller Nc to try.

One can manually select an Nc value that is below the mean search time and repeat Algorithm 1 process with a binary search strategy to find a lower bound. In our case study, the search time number is small as shown in Fig. 16.2; thus, starting with Nc = 1, the manual execution of Algorithm 1 is achievable quickly. For a generic case, where the parameter value is large, this process must be automated by small amount of programming. The found lower bound can be added into Fig. 16.2a to make it more informationally complete, thus co-picture the performance on searching. This lower bound is accurate (or a true lower bound), rather than approximated.

Algorithm 2 sets Nc to a fixed time step number by which it is known the target can be found at least in some configurations and varies the configurations in an attempt to find the exact configuration that allows the target to be found. It utilizes a similar formal verification approach as used by Moulin et al. [25] with additional enhancement to tackle issues from simplifications:

1. Specify system property F "with negation" and verify.
2. If F "with negation" holds, continue to update the configuration, and repeat the check.
3. If the property F "with negation" fails, map the counterexample back to Model 1 and see if it is achievable in Model 1. If achievable, conclude that this exact configuration allows the search to be completed within Nc time. Otherwise, update the system configuration and repeat the check.

In the above, the counterexample of *not* F exactly shows the trace of how the UAVs find the target. Due to approximation, such a trace may not be realizable in Model 1, in which case it must be discarded and other configurations must be further checked. As our map is only 20 $\times$ 20, several different typical configurations (e.g., corner placement, center placement, etc.) that are of interest were tried manually using Algorithm 2 to provide insights on which configurations may be better or optimal. For a more generic case, additional programming may be necessary to automate the process. Note that there may be a chance that all the traces are not realizable. In such a case, as domain experts know the approximation applied, thus under many cases can possibly utilize the trace information to tell what a realizable trace can be, if it exists. In our case study, it happens so that we were able to find realizable traces quickly. The advantage of this algorithm is if it finds the answer, it can save the users from running simulations many times to hit an answer. Even if it does not find the answer, domain experts can still study the traces and may have a chance to get some insights. Under the worst case, one can still use simulations as bottom line.

16.3.3 Formal Verification Results

In this section, to demonstrate the idea of the above approach, we present some interesting experimental results that can be obtained within a reasonable time in the setup we have.

The configuration parameters adjusted in Sect. 16.3.2 are mainly the initial UAV and target placements in most of our experiments. Analysis is performed separately for autonomous and cooperative modes.

An initial set of parameters was assigned to the system randomly in our case study. Specifically, we placed the target at the corner of the cells and set the search time bound as 130 time units. With the cooperative search mode, the verification on the negation of the property F was able to produce results in many parameter settings we tried, in which we can see the traces (i.e., how the target was found) within the time bound, generated from the counterexample data produced by the Design Verifier automatically through Algorithm 2. One of such results is shown in Fig. 16.3; different colors represent different traces of the UAV search paths.

With autonomous mode, it takes an excessively long time (which we didn't wait until finish) for Algorithm 2 to produce verification results, except for some of the settings where the target was placed close to the center of the map, instead of the corners. One of such results is shown in Fig. 16.4, where the target was captured. This shows that the model checker needs more effort to check the settings where the target is placed at the corners. The reason remains to be identified, which is out of the scope of this paper, but is not very important as we already found a configuration where it terminates. This observed phenomenon of course does not mean the autonomous mode has lower performance.

Additional formal verification on the following aspects is also performed using similar methods with traces produced:

- Effectiveness of asynchronous communication protocol
- UAV collisions and live-lock situations

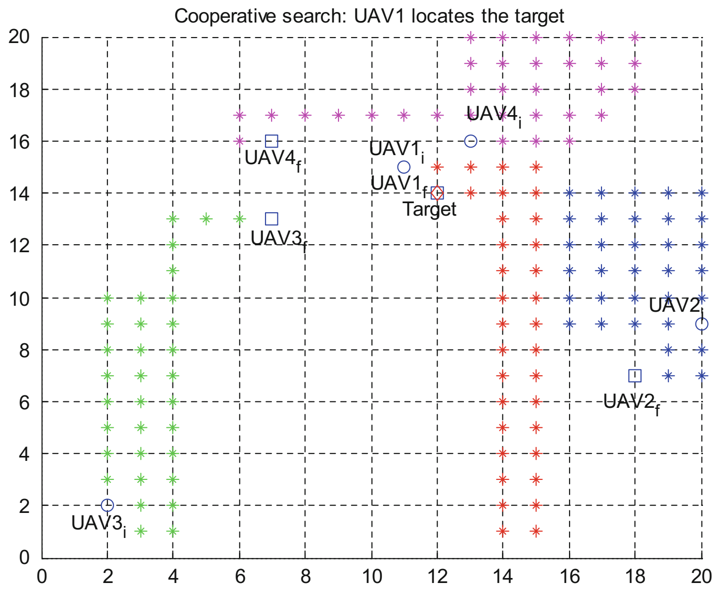


Fig. 16.3 Cooperative search

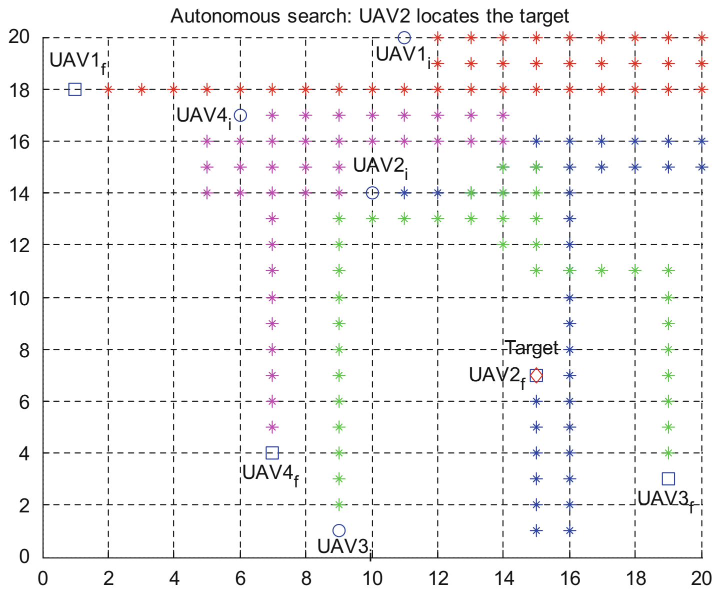


Fig. 16.4 Autonomous search

16.4 Conclusion

Large complex systems such as systems of systems (e.g., Multiple UAV) are difficult to analyze. They are especially difficult to formally verify primarily due to the state space explosion. This paper presents a comprehensive approach on a case study of a public domain UAV planning and search algorithm in two aspects (non-formal and formal) "co-picturing" the system's performance. The approach is able to provide results that cannot be achieved by any of the two aspects alone, especially when the system is too complex that goes beyond the capability of formal verification on the full model. The combination of these two methods provides a comprehensive means to analyze multiple dimensions of a complex system on performance and functional aspects. This process leverages off-the-shelf commercial tools and can be carried out without sophisticated tool development and programming.

Acknowledgments This material is based upon work supported by the US Army, Research, Development and Engineering Command (RDECOM) Contracting Center under Contract No. W911QX-10-C-0073. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the US Army, RDECOM Contracting Center. We would also like to thank Teems Lovett, Andrew Babel and Nick Soldner for their feedbacks on the English language.

References

1. Gaudiano P, Shargel B, Bonabeau E, Clough BT (2003) Swarm intelligence: a new C2 paradigm with an application to control of Swarms of UAVs. In: Eighth ICCRTS Command and Control Research and Technology Symposium
2. Dynasoft project. U.S. Army contract No. W911QX-10-C-0073
3. Zhu H et al (2016) Exploring complex system analysis and verification. In: IEEE International Conference on Automation Science and Engineering (CASE)
4. Sirigineedi G, Tsourdos A, Zbikowski R, White BA (2009) Modeling and verification of multiple UAV mission using SMV. In: Workshop on Formal Methods for Aerospace (FMA)
5. Sirigineedi G, Tsourdos A, White BA, Żbikow R (2011) Kripke modelling and verification of temporal specifications of a multiple UAV system. Annals of Mathematics and Artificial Intelligence
6. Saberi AK, Groote JF, Keshishzadeh S (2013) Analysis of path planning algorithms: a formal verification-based approach. Advances in Artificial Life
7. Maxa JA (2015) Model-driven approach to design a secure routing protocol for UAV Adhoc networks. In: EDSYS, 15ème Congrès des doctorants
8. Konur S, Dixon C, Fisher M (2012) Analysing robot swarm behaviour via probabilistic model checking. Robotics and Autonomous Systems
9. Massink M et al (2012) Analysing robot swarm decision-making with Bio-PEPA. Swarm Intelligence
10. Saha I, et al (2014) Automated composition of motion primitives for multi-robot systems from safe LTL specifications. In: IEEE International Conference on Intelligent Robots and Systems (IROS)

11. Chaki S, Giampapa JA (2013) Probabilistic verification of coordinated multi-robot missions. Model Checking Software
12. Lahijanian M, Andersson SB, Belta C (2012) Temporal logic motion planning and control with probabilistic satisfaction guarantees. IEEE Transactions on Robotics
13. Belta C (2011) Abstractions for planning and control of robotic swarms. Bio-Inspired Computing and Networking
14. Kress-Gazit H, Fainekos GE, Pappas GJ (2007) Where's Waldo? Sensor-based temporal logic motion planning. In: IEEE International Conference on Robotics and Automation
15. Courcoubetis C, Yannakakis M (1988) Verifying temporal properties of finite-state probabilistic programs. In: IEEE Annual Symposium on Foundations of Computer Science
16. Ladowski R (2008) A novel communication protocol using geographic routing for swarming UAVs performing a search mission. Thesis, Air Force Institute of Technology
17. Pack DJ, Mullins BE (2003) Toward finding an Universal search algorithm for Swarm Robots. In: IEEE International Conference on Intelligent Robots and Systems
18. AIMdyn. [Online]. Available: http://www.aimdyn.com
19. Sen K, Viswanathan M, Agha G (2004) Statistical model checking of black-box probabilistic systems. In: Computer aided verification
20. Younes HL (2006) Error control for probabilistic model checking. Verification, model checking, and abstract interpretation
21. Kim M et al (2007) A probabilistic formal analysis approach to cross layer optimization in distributed embedded systems. In: Formal methods for open object-based distributed systems
22. Clarke E et al (2003) Counterexample-guided abstraction refinement for symbolic model checking. JACM 50(5):752–794
23. Hermanns H, Wachter B, Zhang L (2008) Probabilistic cegar. In: Computer aided verification
24. Karp BN (2000) Geographic routing for wireless networks. Ph.D. Thesis, Harvard University
25. Moulin M, Gluhovsky L, Bendersky E (2003) Formal verification of Maneuvering target tracking. In: AIAA Conference on Guidance, Navigation and Control

Chapter 17
A Model Framework for Determining Dynamic Architecture Goals in a Systems-of-Systems

Marc-Andre Chavy-Macdonald, Kazuya Oizumi, and Kazuhiro Aoyama

Abstract This paper presents results from a prototype modeling methodology, part of a PhD project intending to create an integrated model framework relating societal modeling techniques to system-of-systems (SoS) architecture models and design. This framework uses matrix-based and system dynamics (SD) models to relate product system functions to associated systems and to societal dynamics step-by-step, in order to synthesize justified design priorities for a new product platform. In this manner, function importance weights or goal prioritization for a new product are obtained from societal models and scenarios. These weights are highly dependent on outside (SoS) functions, vary in time, and depend on the risk attitude of the project. In future work, this framework will be applied to modularization and architecture design and evaluated by a case study in the aerospace industry, where a space system manufacturer is considering designing new satellite product lines.

Keywords Systems engineering • Product design • Systems-of-systems • Sociotechnical systems • Complexity management • Design structure matrix • System Dynamics • Functional analysis

17.1 Introduction

Systems-of-systems (SoS), product service systems (PSS), product ecosystems, complexes, and collections have in common that they are generally about the environment of a given system or product, or “one level up” in the hierarchy from the observer’s standpoint [1], though their specific characteristics differ. When defining a system architecture, it is very useful to “look one level up,” as the criteria for a product’s success may reside in its match with its environment—increasingly so in an ever-more dynamic and interconnected world. The young field of SoS engineering is in need of research to extend its modeling techniques to suit the particular needs of such systems, centered around dynamics, autonomous but

M.-A. Chavy-Macdonald (✉) • K. Oizumi • K. Aoyama
University of Tokyo, Tokyo, Japan
e-mail: marc@m.sys.t.u-tokyo.ac.jp

interacting systems and emergent functionality [2, 3]. Perhaps equivalently, it also needs extension to include sociotechnical aspects, or systems with societal actors or implications [4].

A central purpose of system architecture is defining the goal and hierarchy of goals of a system. Indeed, goals consistency and prioritization are “at the heart of systems engineering” [1], trying to answer the key question “are we building the right system?”. In new product development, it is often quite unclear how to define system goals, the importance of product functions, and what should be the design target. This is often due to “market complexity,” or at least societal complexity, residing one level up in the hierarchy. This complexity is marked by dynamism, as in SoS, and so the question of goal and function prioritization must also consider change management.

The research question of the project that this paper initiates is *how to design a new product platform considering its dynamic stakeholder environment*, such as a SoS. We seek to create an integrated modeling framework for decision support in architecting such a product considering societal dynamics and have selected matrix-based methods and system dynamics (SD) as a starting point.

17.2 Relevant Literature

Matrix-based methods such as design structure matrices (DSMs), multiple domain matrices (MDM), and QFD serve to concisely represent graph-based product (or process, or organization) information, such as components, functions, or requirements, and may be most useful when used together with graph views [5]. These methods need extension to dynamic phenomena, as well as work on the goals domain and objective functions for modularization, according to the study by Browning [6].

In previous work, matrix and graph methods were used to consider product, design process, and organization simultaneously [7]; here, we focus on the fourth of Lindeman et al.’s four domains of complexity, and the root one: market complexity, which is also dynamic. Some recent work has tried to extend matrix-based methods to dynamic phenomena, in particular in the project domain [8], and more recently in the case of measuring PSS dynamics [9], both using system dynamics. Other recent work has extended DSM usage to sociotechnical systems using functional analysis [10].

To model qualitative, societal, and highly dynamic phenomena, we also select System Dynamics (SD) [11]. SD offers the advantages of promoting conceptual understanding, with an emphasis on visuals and fundamental system dynamic structure, which appears promising for the uncertainty inherent in architecting. It also has a following in modeling business and policy, domains of interest for SoS.

Systems thinking bridges multiple disciplines, with many viewpoints studying similar phenomena. As such, for our first toy model, we choose the case study by Shove and Southerton [12], on the gradual arrival of complementary goods and arising of a whole system around the household freezer and its usage.

17.3 Methodology

A global overview of the project's proposed modeling methodology is shown in Fig. 17.1. Starting from a product segment map [13] and tentative plan, we use a two-phase modeling process: first SD modeling defines the surrounding dynamic societal system and so the designer's goal, in terms of functions and their priorities; the second is matrix-based modeling utilizing these priorities to define a product platform and modules. The end goal is the architecture of a platform, modules, and product plan.

The right part of Fig. 17.2 shows the specific methodology demonstrated in this paper, which is the detail of the boxed part of Fig. 17.1. The first three steps are modifications of the procedure explained in detail in the study by Lee et al. [9]. From a partial architecture or system elements, we perform a functional analysis at a high abstraction level and map interconnections, creating a functional dependency network, similarly to the study by Lee et al. [9], and representable in a DSM. However, by augmenting this network to include associated actors, systems, and products, as in the study by Vaishnav et al. [10], we may obtain a larger picture of the surrounding SoS. These functions may be mapped to functional measures, representable in a causal loop diagram (CLD) in System Dynamics [9]. This allows an understanding of the principal dynamics of the system surrounding the product of interest. In Step 3, we may populate the CLD to be a fully executable System Dynamics model and simulate, allowing exploration of scenarios and sensitivity of net present value (NPV) to different functions. These sensitivities may be used to derive importance weights for each function, prioritizing them. These importance factors will be time dependent, since the sensitivities will vary with SoS configuration, and hence can be used to define time-dependent, justified importances. This paper ends here, leaving full architecture definition (last box) for further work. The following subsections will explain Steps 1–7 of Fig. 17.2 in detail, and some results.

17.3.1 Evolving, Sociotechnical Functional Dependency Network

Following the study by Lee et al. [9], in Step 1, the elements and functions of the product and associated systems are identified, and dependencies between functions are assessed. Functional dependency was assessed using the same question as in the

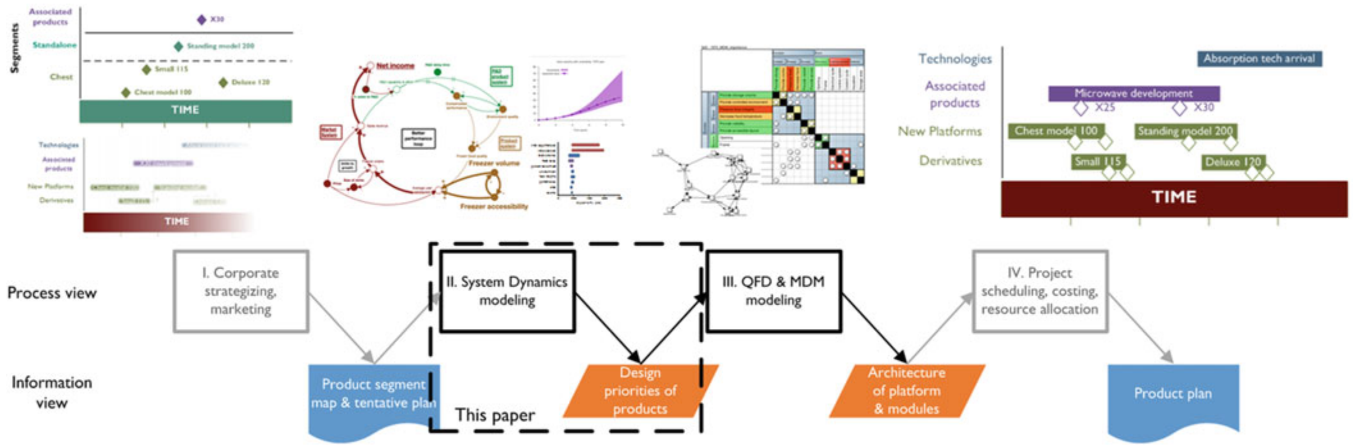


Fig. 17.1 Overarching research plan, concerned with defining Boxes II and III and their links. This paper's scope is phase II (*boxed*): System Dynamics modeling. Endpoints are product segment plan (*left*) and full product plan and architecture (*right*) [13]

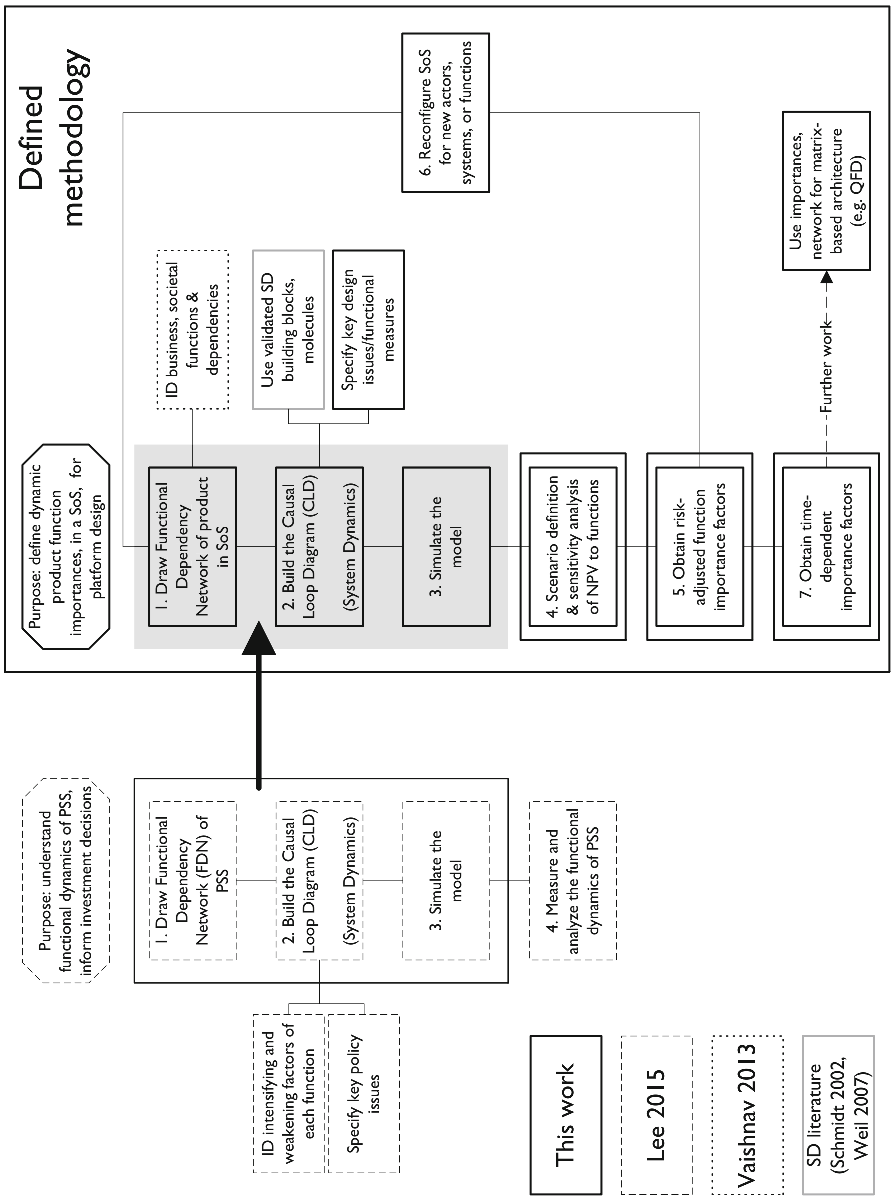

Defined methodology
Purpose: define dynamic product function importances, in a SoS, for platform design
1. Draw Functional Dependency Network of product in SoS
2. Build the Causal Loop Diagram (CLD) (System Dynamics)
3. Simulate the model
4. Scenario definition & sensitivity analysis of NPV to functions
5. Obtain risk-adjusted function importance factors
7. Obtain time-dependent importance factors
ID business, societal functions & dependencies
Use validated SD building blocks, molecules
Specify key design issues/functional measures
6. Reconfigure SoS for new actors, systems, or functions
Further work
Use importances, network for matrix-based architecture (e.g. QFD)
Purpose: understand functional dynamics of PSS, inform investment decisions
1. Draw Functional Dependency Network (FDN) of PSS
2. Build the Causal Loop Diagram (CLD) (System Dynamics)
3. Simulate the model
4. Measure and analyze the functional dynamics of PSS
ID intensifying and weakening factors of each function
Specify key policy issues
This work
Lee 2015
Vaishnav 2013
SD literature (Schmidt 2002, Weil 2007)

Fig. 17.2 The methodology of this paper, in frame on *right*, and relation to literature. Most useful outputs are *double-boxed*

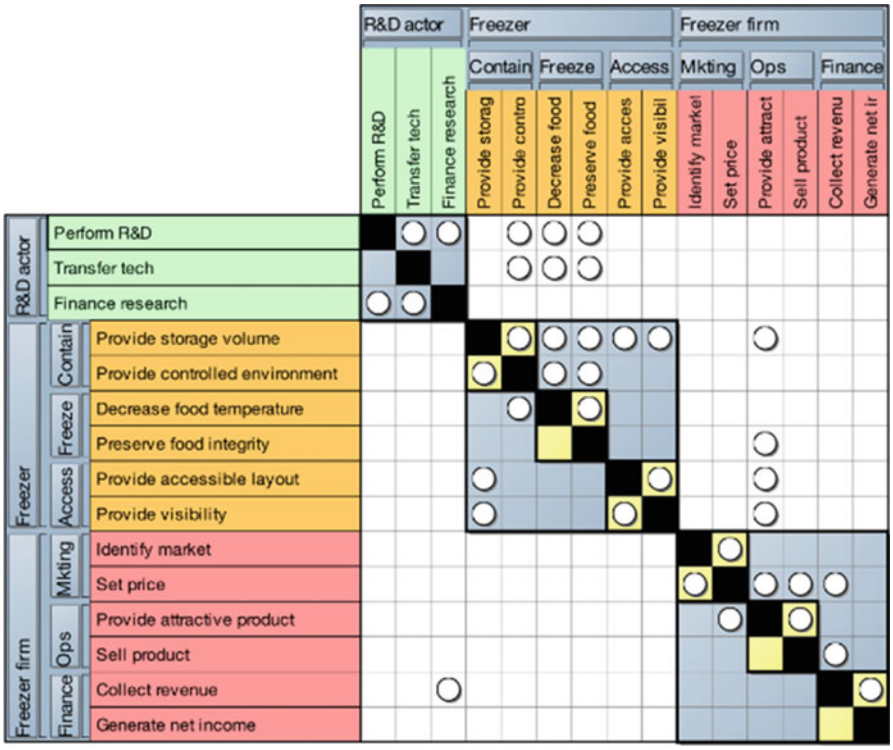


Fig. 17.3 The functional interaction matrix for the first period of the studied SoS. Systems and their functions are on the *left*

study by Vaishnav et al. [10]: *Is Function A necessary to fully or partially perform Function B*? If so, B (on the vertical) depends on A (horizontal). The result for the freezer, freezer firm, and R&D actor is shown in Fig. 17.3, made with the CAM software [14]. In terms of assessing dependency, we note the difficulty of avoiding mapping indirect connections, especially in the functional domain, and the usefulness of the matrix format for info acquisition.

Figure 17.4 illustrates the evolution of the functional network defined above; as described in the study by Shove and Southerton [12], we may define three distinct phases of product evolution (colored horizontal bands), and each may be described by a functional network, here in matrix form. Following time downwards on the y-axis, we note the arrival of new functions (arrows) and new actors or systems with strong interactions with the freezer product. The use case (dotted arrows) changes gradually. Maier's characteristics are perhaps the most common SoS "definition" [3], and this system appears to qualify [2]: the constituents are certainly autonomous, independent systems with their own purpose and are geographically distributed and evolving. From the user perspective, an emergent function arises in Phase 3: *providing frozen meals,* dependent on interactions. The dynamics of Fig. 17.4

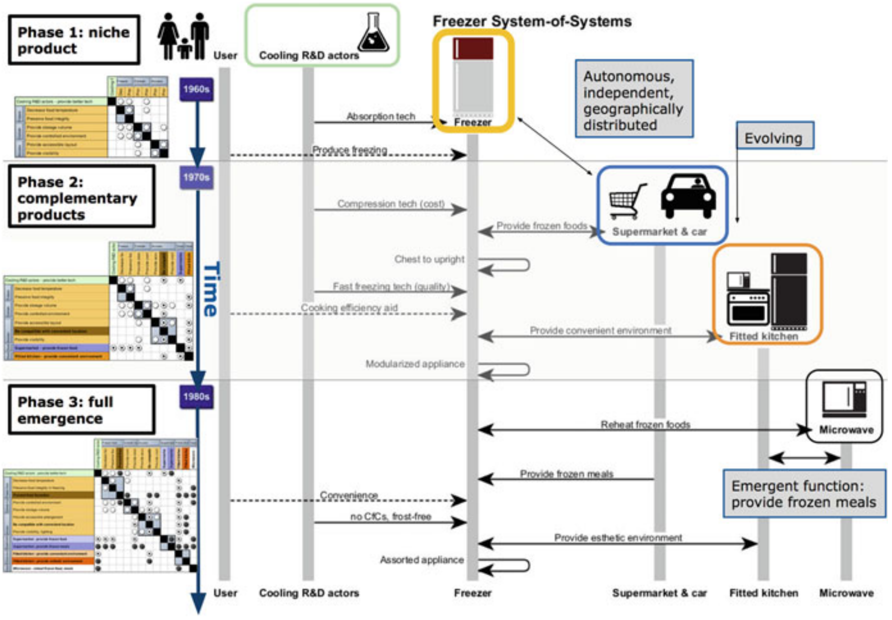


Fig. 17.4 Overview of the evolving SoS in three phases, with arrival of functional interactions and systems reconfiguring the network

applies to graph reconfiguration and new networks via the arrival or disappearance of functions, actors, or systems; to discuss other dynamics, we use System Dynamics.

17.3.2 Causal Loop Diagram

Step 2, following the method of Lee et al. [9], is to map the functional dependency network from Fig. 17.3 to a causal loop diagram (CLD) in System Dynamics, done here with the software Stella Professional 1.1.2. This is done by mapping functions to functional measures and dependencies to directions of change. To obtain a relatively understandable CLD, we neglect several functional interactions deemed less important, as shown in Fig. 17.5, and also augment with additional factors [9]. However, since our purpose is architecture synthesis, unlike the study by Lee et al. [9], we augment the product functions with those of the firm, allowing some usage of existing schema from system dynamics (e.g., in [11], [15], or [16]). In reality, the model creation may not be linear but iterative in Steps 1 and 2, and the CLD creation may spur the process of adding variables or functions, perhaps missing on the business side. Importantly, this CLD step is a further check on the

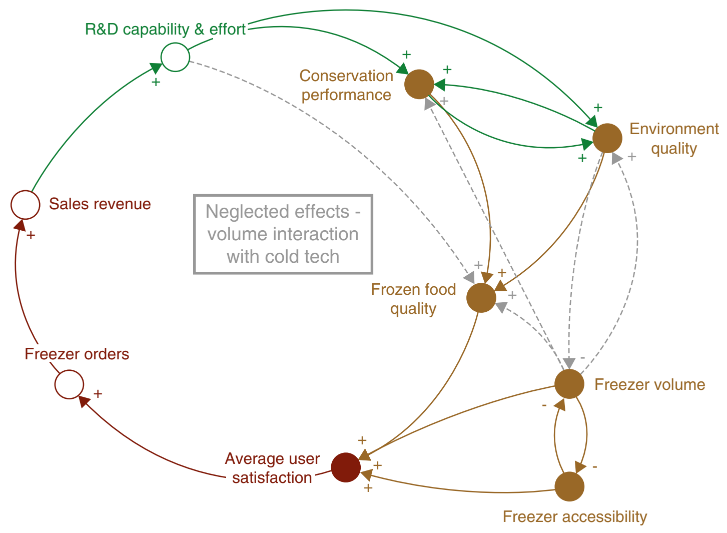


Fig. 17.5 Intermediate step in forming CLD: neglecting some functions, finding a need for more nodes (*red*)

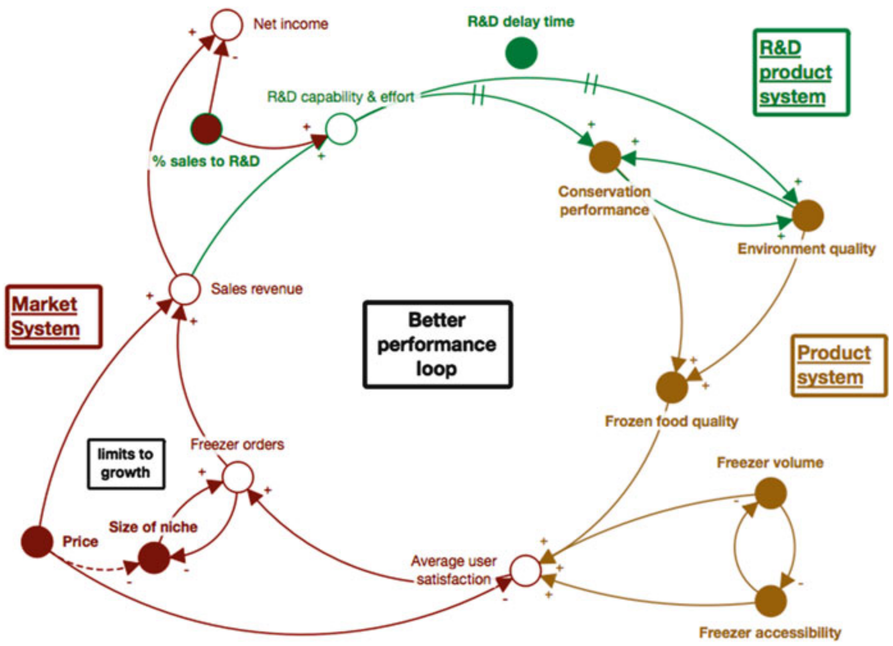


Fig. 17.6 A CLD of the first SoS period, showing main dynamics. The main feature is the central reinforcing loop

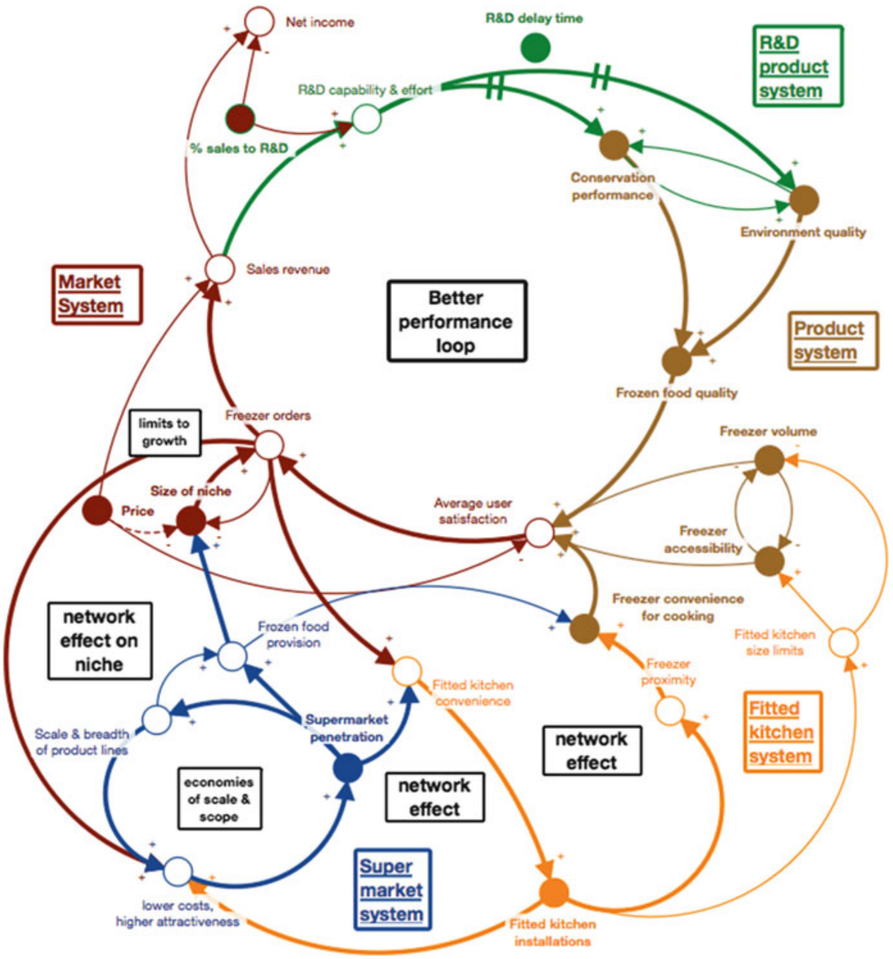


Fig. 17.7 A CLD of the second phase of the example SoS. *Thicker arrows* indicate the main feedback loops

matrix, especially useful for the ambiguity of high-level functions "necessary in part" for others.

A key point is obtaining a measure of effectiveness for the product from a societal viewpoint, which for the corporate case is basically expected net present value (NPV), perhaps tempered by strategic considerations and risk appetite. The result is shown in Fig. 17.6. This CLD is very useful for qualitative understanding of the dynamics. We note that there are three interacting systems distinguished by colors (corresponding to Fig. 17.3); their dominant interaction is a large central loop, the "new product development" or "better performance" loop, characterized by Sterman as one of the positive feedbacks at the root of corporate growth [11]. Market success spurs R&D investment, leading to an improved product and further success. A balancing loop occurs due to the limited size of the niche of potential customers.

Likewise, Fig. 17.7 is a key to discussion of the second SoS phase ("complementary products" in Fig. 17.4), covering approximately the period of 1970–1980. New actors/systems and associated functions have become important, leading to more complex dynamics. We may clearly see the loop from Fig. 17.6 at the top. Two associated systems, or actors supplying complementary products, have arrived: the supermarket, and the fitted kitchen and its providers. The main dynamics engendered are network effects: fitted kitchen penetration increases freezer attractiveness and vice versa; however, it also imposes requirements on the size and shape of the freezer as well as demands a new product function: compatibility with the kitchen and cooking, measured here by *convenience for cooking*. The supermarket, on the other hand, has a different type of network effect: increasing the size of the niche of potential customers, from conservers of garden produce to buyers of frozen supermarket foods. Finally, network effects exist between the supermarket and fitted kitchen, as each increases the other's attractiveness.

17.3.3 System Dynamics model

Step 3 is to transform the above CLDs into full executable System Dynamics models. Unlike Lee et al. [9], we do not add "intensifying or weakening factors," but rather break down to finer granularity, utilizing similar schema as in the models of [11, 15, 16] and other SD literature. The result is the SD model shown partly in Fig. 17.8 of the early SoS (Phase 1 of Figs. 17.4 and 17.6). In it, such variables as product attractiveness, sales, and NPV can be tracked dynamically over long periods in response to changes in the functional measures of the product, or of associated systems. A summary of the main interactions of the later SoS (as explained in Fig. 17.7) is shown in Fig. 17.9. We note the interacting systems, main dynamics, and interactions modeled. The main systems are the product, its market, R&D for the product, and two complementary products: the supermarket and fitted kitchen. All interact simultaneously.

17.3.4 Sensitivity and Scenarios

Step 4, in a partial departure from the study by Lee et al. [9], is not only the definition of scenarios but sensitivity analysis of NPV. Scenarios are defined by different values of functional measures in the SD model. After determining a "baseline" of plausible values and results, the effect of altering functional measures can be explored. Some results of the System Dynamics of product diffusion can be seen in Fig. 17.10. The left part of the figure depicts cumulative NPV for several scenarios, while the right part shows the intermediate calculations, sales, and

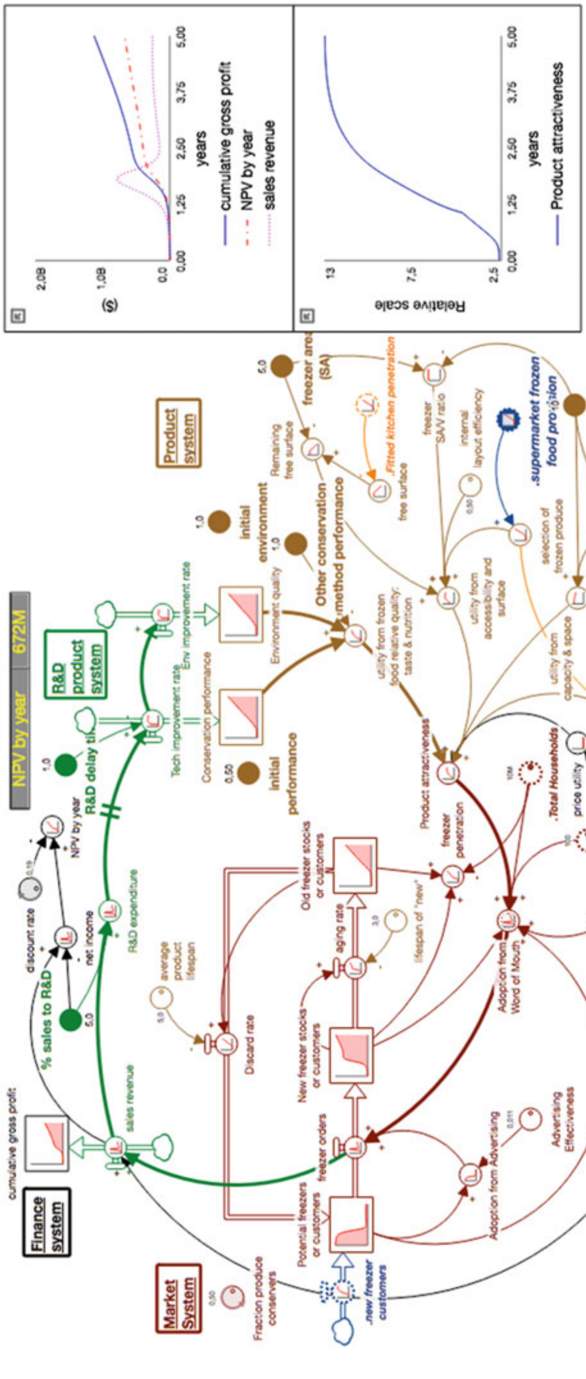


Fig. 17.8 Full SD model of Fig. 17.6. Note the product key parameters and functional measures (*filled nodes*) linked to the SoS

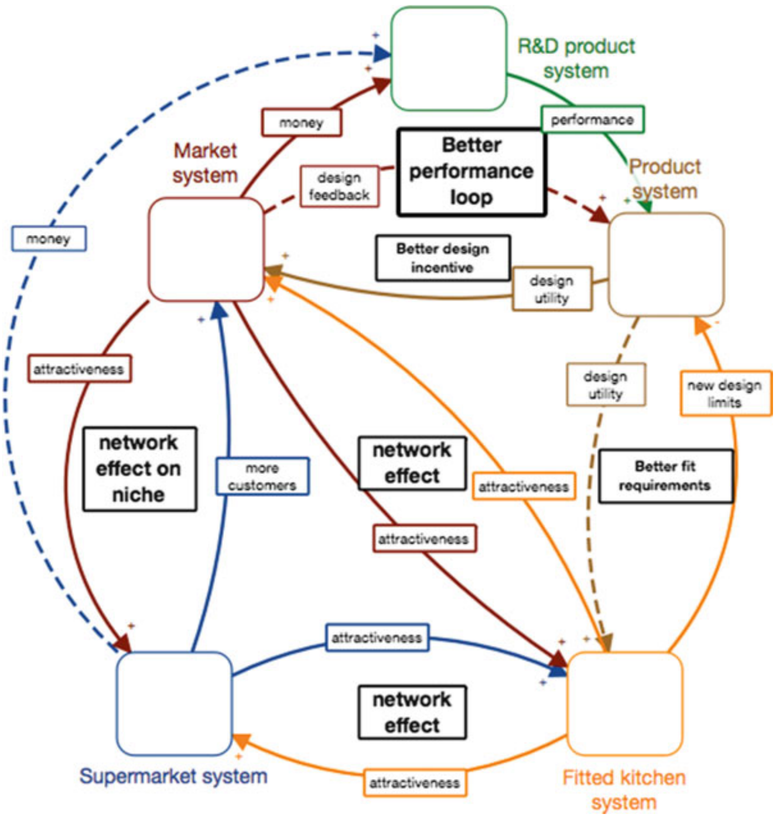


Fig. 17.9 A systems view of the Phase 2 SoS, as modeled by our SD model. The *dotted arrows* are neglected interactions

product attractiveness evolution, during the same period. Illustrating the dynamics of Fig. 17.6, the light blue curve is the baseline design, while the yellow curve depicts higher freezing technology performance, with shading showing the uncertainty for each. Uncertainty is estimated hereby running may "small" variations in functional measure value, and observing the maximum deviation. The orange scenario shows a slower R&D cycle. Note that, e.g., technology differences here may yield substantially different initial product attractiveness but that due to the dominance of the R&D dynamic (and small niche size) over the longer term, initially higher sales have very little effect on eventual NPV. On the other hand, a change in the dynamics, here a slower R&D transfer, yields a large difference in NPV, despite an initially successful product (high attractiveness and sales). Thus, certain functions within the product and SoS are far more important than others for long-term NPV, and these must be understood and prioritized. Larger differences in dynamics can be seen in Fig. 17.7, shown by the dark blue *1980* curves. We note

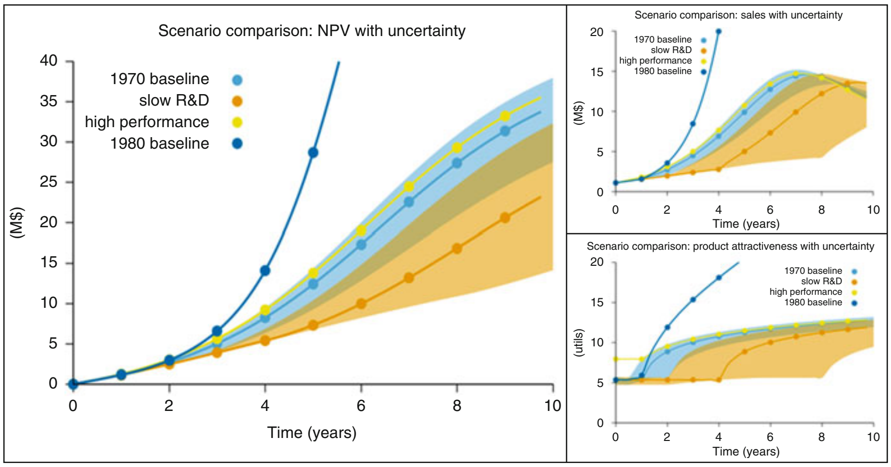


Fig. 17.10 SD-simulated scenarios—cumulative NPV (*left*), corresponding sales (*top right*), and product attractiveness (*bottom right*) based on different function performances. The shaded area shows uncertainty, determined by varying many parameters at once

that a market for (let alone *prior presence of*) complementary goods greatly accelerates diffusion.

The scenarios shown in Fig. 17.10 contribute to qualitative understanding of the effect of different functional measure values. They also aid in defining a “large but plausible range of input variation” for each functional measure in the SoS, which can be used for sensitivity analysis of NPV. Results are shown in tornado plots in Fig. 17.11. The blue bars correspond to product functions, the red ones to functions of other SoS actors, normally outside the architect’s control, and the purple ones to associated functions which may or may not be within control or influence, depending on the organizational structure. Period 1 corresponds to Figs. 17.3 and 17.6, while Period 2 shows results for the evolved SoS depicted in Fig. 17.7, with the arrival of other systems. Immediately noticeable is the outsize impact of complementary goods or associated products in Period 2. Sometimes the SoS function most determining product commercial success is beyond design control. This may have implications for the target market, or any design at the system interfaces. Also, perhaps the product team strategy should be to obtain more information regarding other SoS systems, form partnerships or agreements, or update the risk profile or deployment approach. We also note that even between the “designable” blue and purple functions, there are differences in expected impact on NPV. This can inform the system architecture and design strategy. Sometimes more “balanced design” is good (left tornado, mostly), but other times it would be a mistake not to highly prioritize functions (right tornado).

17.3.5 Obtain Importance Factors and Iterate

Step 5 is to transform the obtained sensitivities of NPV to functional measures, to importance factors of functions for design. Importance weighting should consider the risk appetite of the project and organization. To illustrate, we consider three categories: risk-averse, risk-neutral, and risk-taker projects. For the case of the risk-neutral project, weights should be based on a simple ratio of sensitivities; thus, weighting functions proportionally to their range of effect on NPV (width of the tornadoes of Fig. 17.11). The middle bar stack of the left-hand side of Fig. 17.12 illustrates results: we obtain that 32% of *all* variation in NPV due to functional measures is explained by one function's impact. We assign it an importance weight of 32%. Note that two functions together—R&D transfer and food quality—explain half the variation in value. Other than these functions which may be thought of as "critical," a good performance of other functions can be classified broadly as "important" or merely "desirable" system goals [1], according to their potential impact on NPV.

In the case of a risk-averse project or organization—if nonfailure is required and a smashing success not a huge priority (e.g., projects with geopolitical aspects or involving human safety)—function importance should be based on weights designed to "minimize potential damage" to NPV. We weight functions proportionally to the minimum NPV they may allow, reading from the left edge of the tornados in Fig. 17.11. The left part of Fig. 17.12 shows results for such projects.

A risk-taker project—for which failure is an option and a mediocre success undesirable (e.g., smaller projects in highly competitive, fast-paced markets, like web apps)—might prefer another metric. We weight functions proportionally to their highest payoff to NPV: the rightmost edge of the tornadoes of Fig. 17.11.

Step 6 is to reconfigure the network for the next SoS phase, redefining the systems and functions, as seen in Fig. 17.4. Going through Steps 1–5 again will yield a different CLD (e.g., "1980 case", Fig. 17.7), scenarios and set of importance factors, to be used in Step 7: dynamic importances, on the right of Fig. 17.12. Thus,

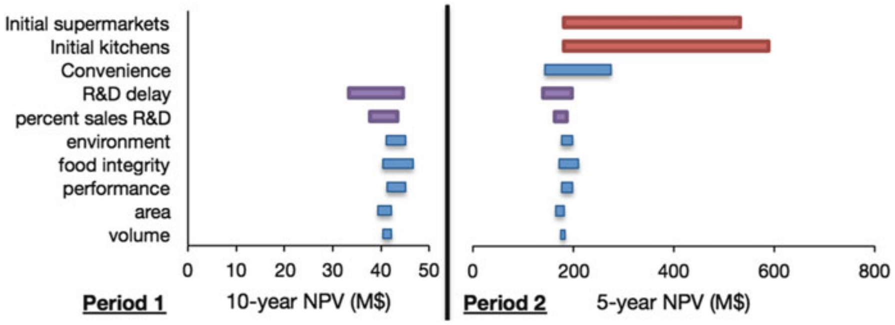


Fig. 17.11 Sensitivities of NPV to SoS function performances—*red* are outside control. *Left*: 1960–1970 case, *right*: 1970–1980 case

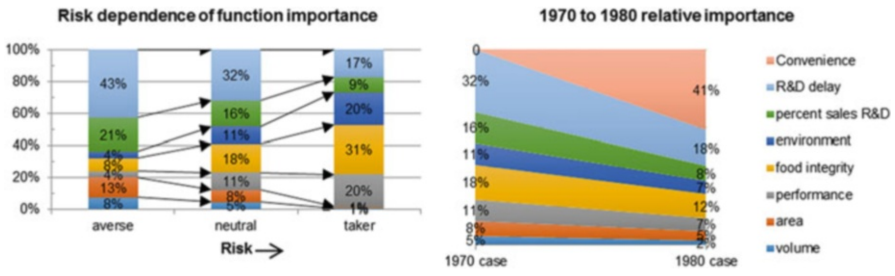


Fig. 17.12 Derived importance weights of different functions vs. risk appetite (*left*) and time (*right*): critical for change management

crucially, "ideal" weights vary in the dynamic SoS: the system goal is a moving target. This might inform platform strategies and change management. In particular, periods of maximum change in "critical" functional metrics for NPV are those in which the product should change, preferably affordably via, e.g., module changes. Good module candidates might group functions that undergo rapid, simultaneous increase in importance.

17.4 Discussion and Toward Architecture

The intended use of such a model is as a transparent, group decision support system, not for stand-alone predictions. One main finding of such modeling is that the voice of the customer, either directly or via marketing, is not enough to determine system goals or function importance. Nor is taking into account all stakeholders or company capabilities; instead, we need dynamic, holistic modeling of their combined impact on the business. Only by integrating these perspectives—in particular, looking at *long-run* effect on business value of these factors, in interaction with product functions, can we determine system goals and importance weights. The end purpose must be value, or expected NPV. To limit its inaccuracy, such a model should input conjoint analysis, common in marketing, for product attractiveness modeling [15].

The authors now think that a primary characteristic of system goals, or high-level product functions, is that the "ideal value" changes with time. This comes from the dynamics of society, which are faster than most product lifetimes. Thus static, "precise" multidisciplinary optimizations may miss the point, even if they target NPV with very accurate product models and consumer preferences. This is because we have an uncertain, but especially *varying* objective function.

The natural next step is to create a full modularization plan and system architecture using derived weights, on a real case study, as seen in the last box of Fig. 17.2. We should obtain the platform architecture and product plan (see Fig. 17.1). Key topics include combining the above with formal scenario analysis, a powerful technique which might complement and be suitable for both defensible business decisions and qualitative phenomena. The large uncertainties present also means the value of real options is higher. This is also the case for expensive systems, such as satellites. We will consider both in future work.

17.5 Conclusion

In summary, we have presented a simple model linking functional metrics to business and societal values, with an emphasis on (a) decision support and a transparent model and decisions, (b) many stakeholders to populate the model, (c) NPV as the correct design target, (d) project risk appetite as critical, and (e) high uncertainty and dynamics. When we "look upwards" in the system hierarchy at the market and surrounding systems, we find that other actors, systems, and products may be more important for success than some main product functional measures. Additionally, the dynamic behavior of the system or SoS as a whole may be more important than traditional design priorities, such as product attractiveness. System architects should understand this and rationally focus on those factors deemed to matter more.

In applications, this method might often come up with a "null result": with uncertainty too high, it is unclear what to do. This methodology and SD model might then serve a useful purpose by showing *where* information is needed: in consumer preferences, or some societal actor, or technology forecasting ... because of the link to NPV, we get a direct "value of information" calculation. It might also signal in what area real options are needed, to move forward. Although only early phase, we think this line of inquiry and modeling methodology is rich in possibilities.

References

1. Crawley E, Cameron B, Selva D (2015) System architecture: strategy and product development for complex systems, 1st edn. Prentice Hall, Boston
2. Maier MW (1996) Architecting principles for systems-of-systems. In: INCOSE International Symposium, 6:565–573, Wiley Online Library
3. Nielsen CB, Larsen PG, Fitzgerald J, Woodcock J, Peleska J (2015) Systems of systems engineering: basic concepts, model-based techniques, and research directions. ACM Comput Surv 48:18:1–18:41. doi:10.1145/2794381
4. SEBoK. http://sebokwiki.org/wiki/Guide_to_the_Systems_Engineering_Body_of_Knowledge_(SEBoK). Accessed 14 Feb 2016
5. Lindemann U, Maurer M, Braun T (2009) Structural complexity management. Springer, Berlin/Heidelberg
6. Browning TR (2016) Design structure matrix extensions and innovations: a survey and new opportunities. IEEE Trans Eng Manag 63:27–52. doi:10.1109/TEM.2015.2491283

7. Oizumi K, Aoyama K (2013) Design orchestration composer–a model base enabling holistic management of product, design process, and organization. In: DS 75-3:design for harmonies, Vol. 3: Design organisation and management, Seoul
8. Kasperek D, Maisenbacher S, Maurer M (2014) Structure–based compilation of system dynamics models for assessing engineering design process behavior. In International Systems Dynamics Conference, pp 233–242
9. Lee S, Han W, Park Y (2015) Measuring the functional dynamics of product-service system: a system dynamics approach. Comput Ind Eng 80:159–170
10. Vaishnav C, Choucri N, Clark D (2013) Cyber international relations as an integrated system. Environ Syst Decis 33:561–576
11. Sterman J (2000) Business dynamics: systems thinking and modeling for a complex world. McGraw-Hill Education, Boston
12. Shove E, Southerton D (2000) Defrosting the freezer: from novelty to convenience anarrative of normalization. J Mater Cult 5:301–319
13. Ulrich KT, Eppinger SD (2007) Product design and development, 4th edn. McGraw-Hill Higher Education, New York/London
14. Wynn, DC, Wyatt DF, Nair SMT, Clarkson PJ (2010) An introduction to the Cambridge advanced modeller. In: Proceedings of the 1st international conference on modelling and management of engineering processes, Cambridge, UK

15 Schmidt MJ, Gary MS (2002) Combining system dynamics and conjoint analysis for strategic decision making with an automotive high-tech SME. Syst Dyn Rev 18:359–379

16 Weil HB (2007) Application of system dynamics to corporate strategy: an evolution of issues and frameworks. Syst Dyn Rev 23:137–156

Chapter 18
Understanding How Social Network Analysis Can Provide Insights Into Emergent Networks of Systems

James R. Enos and Roshanak Nilchiani

Abstract The Department of Defense (DoD) is currently faced with a major challenge as individually designed and procured systems become more interoperable and dependent upon each other to provide value to the warfighter. The DoD currently has 79 Major Defense Acquisitions Programs (MDAP) in addition to the thousands of smaller programs and legacy systems that all operate on the battlefield together. DoD Systems engineers must understand how these systems interoperate on the battlefield to deliver the desired effects for the warfighter. However, traditional systems engineering and system of systems analysis tools and methods are inadequate to describe, visualize, and analyze the complex network of systems that have emerged over the past several decades. There is an opportunity to apply and adapt social network analysis (SNA) tools and methods to systems to understand the network, quantify the "ilities," and anticipate the effect of changes to the network. This chapter examines systems engineering, systems architecture, and systems of systems (SoS) analysis to identify shortfalls in the method to deal with emergent networks of systems. It proposes a combination of individual system architectures to develop the network of systems. It goes on to understand how SNA tools and methods can be adapted to understand networks of systems and identify the potential for the quantification of some of the systems engineering "ilities" through SNA metrics. Finally, it discusses a proposed research path forward to further apply these tools to a more robust representation of the DoD network of systems, further analyze the potential application of SNA tools to networks of systems, and a method to validate and verify the results of this work.

Keywords Social network analysis • Systems engineering • System architecture

J.R. Enos (✉) • R. Nilchiani
Stevens Institute of Technology, Hoboken, NJ, USA
e-mail: jenos@stevens.edu; rnilchia@stevens.edu

A.M. Madni et al. (eds.), *Disciplinary Convergence in Systems Engineering Research*, https://doi.org/10.1007/978-3-319-62217-0_18

Nomenclature

C_{A}	Group centrality
$C_{\mathrm{b}}(n_i)$	Betweenness centrality of node i
$C'_{\mathrm{c}}(n_i)$	Closeness centrality of node i
$C'_{\mathrm{d}}(n_i)$	Degree centrality of node i
D	Network density
$d(n_i)$	Connections for node i
$d(n_i, n_j)$	Connections between node i and j
g	Total nodes in network
g_{jk}	Edges between nodes j and k
n_i	Node i
N_{s}	Subgraph of network N

18.1 Introduction

The Department of Defense (DoD) has been successful in using social network analysis to analyze, visualize, and understand complex networks of terrorists and individuals of interest in both Iraq and Afghanistan. Most publicized was the DoD and the Central Intelligence Agency's (CIA) use of social network analysis to identify the location of Usama Bin Laden. For years, special operations forces searched the mountains of Afghanistan and Pakistan trying to locate the most wanted man in the world. It was not until a CIA analyst identified a courier within the Bin Laden network that the DoD was able to pinpoint his location and execute a daring night raid against the leader of al-Qaeda [1]. However, the DoD has not used these methods and techniques to look inward at their complex network of systems to better understand how their network of systems has emerged over the past several decades. These systems operate on the battlefield; however, traditional systems engineering and system of systems analysis tools and methods are inadequate to describe, visualize, and analyze this complex network of systems. There is an opportunity to apply and adapt social network analysis (SNA) tools and methods to systems to understand the network, quantify the "ilities," and anticipate the effect of changes to the network. This chapter examines systems engineering, systems architecture, and systems of systems (SoS) analysis to identify shortfalls in the method to deal with emergent networks of systems. It proposes a combination of individual system architectures to develop the network of systems. It goes on to understand how SNA tools and methods can be adapted to understand networks of systems and identify the potential for the quantification of some of the "ilities" through SNA metrics.

18.1.1 Complexity of the Department of Defense Network of Systems

The DoD faces challenges in the integration and management of the network of systems that it has developed over the past 30 years. In 1996, the Vice Chairman of the Joint Chiefs of Staff proposed warfighting capability would be more reliant on systems of systems (SoS) and network-centric operations [2]. As such, DoD systems are becoming more and more interconnected and reliant on other systems to provide capability to the warfighter. Often, individual systems on a battlefield cross-service boundaries making collaboration difficult in traditionally hierarchal military structures [3]. In addition, the GAO found that the DoD lacked methods and tools for conducting portfolio management at the enterprise level and noted that there were gaps in the DoD's ability to identify, understand, and assess the capability portfolio [4].

The DoD manages warfighting capabilities with the Joint Capability Integration and Development System (JCIDS) process that was designed to ensure that validated military capability requirements drove acquisitions. The JCIDS process requires sponsors to generate three main documents: Initial Capability Document (ICD), Capability Development Document (CDD), and the Capability Production Document (CPD). Each of these documents support different phases in the development and acquisition process by providing traceability from warfighter capability requirements to fielded systems [5]. In addition, the joint staff requires several DoD Architecture Framework (DoDAF) viewpoints to support the development of warfighter capabilities. System architects use DoDAF to capture multiple perspectives of a warfighting capability's system architecture. One shortfall of the DoDAF architectures used in capability development is that they do not capture a DoD-wide perspective of the interactions between individual systems. In an effort to understand shortfalls in DoDAF, the National Defense Industrial Association's Systems Engineering Division charted an Architecture Frameworks Working Group and an initial recommendation was to standardize architectural elements and modeling methodology to integration of architectural data [6]. Ring, et al. proposed another effort to aggregate DoDAF architectures, that is, the Activity-Based Methodology that integrated architectures by the organization, system, and role aspects of the architecture [7]. A third effort proposed aggregating independent architectures through a system, capability, and mission perspective by utilizing independent DoDAF viewpoints [8].

18.2 Literature Review

This section presents a brief review of the relevant systems engineering literature that provides a foundation for the exploration of emergent networks of systems. It examines systems engineering, systems of systems and family of systems analysis, and systems architecture. These are all methods systems engineers use to deal with the complex nature of current systems and the interactions between systems. However, there is an opportunity to contribute to the literature by closing the identified gap in the analysis of interactions between systems.

18.2.1 Systems Engineering

Systems engineering as a discipline is faced with increasing complexity of systems as technology progress and systems are more interconnected. International Council on Systems Engineering (INCOSE) defines systems engineering as "an interdisciplinary approach and means to enable the realization of successful systems [9]. It focuses on defining customer needs and required functionality early in the development cycle, documenting requirements, and then proceeding with design synthesis and system validation while considering the complete problem." Systems engineers differ from traditional engineers in that they consider the system in its entirety, lead the conceptual design of systems, and bridge the gaps between traditional engineering [10]. As such, systems engineers have developed a variety of means: system architecture, system of systems, and enterprise architecture to deal with complexity.

Systems engineering emerged from the early age of the telephone industry in the 1920s and 1930s when telecommunications companies identified a need for a multidiscipline approach to the design of their systems [11]. During World War II, systems thinking and operations research emerged as disciplines to improve military systems. After the war, the DoD furthered these efforts to design more complex and advanced systems. Soon after the war, Goode and Machol published the first systems engineering text that introduced the methodology and provided a basis for further research [12]. The field developed between the 1960s and 1990s, and systems engineers founded INCOSE in 1990 [13].

As a foundation, systems engineering translates customer and stakeholder needs into system requirements and allocates these requirements to physical system during the design of a system [14]. Systems engineering has progressed significantly over the last century, but it remains focused internally to a system. Systems engineers decompose a system of interest, both functionally and physically, design these decomposed subsystems and components, then combining them to achieve the overall function of the system. Systems engineers have adopted the "ilities" as a construct for understanding nonfunctional attributes of systems and the complex interactions between systems. Several "ilities" capture a system's ability to change and may be quantifiable through the use of social network analysis metrics. Specifically, robustness deals with the system's ability to maintain a level of performance regardless of the environment and situation, versatility is the ability of a system to fulfill a variety of stakeholder expectations, and interoperability is the system's ability to operate among other systems [15].

Within the field of systems engineering, there is an opportunity to better understand the complexity of systems, systems of systems, and enterprise systems. In 2006, a workshop consisting of thought leaders from a variety of disciplines met to discuss the issue of complex systems, and one area that received substantial attention was the modeling of complex systems with an emphasis on the dynamic, networked nature of systems [16]. Both internal to the DoD and with the broader systems engineering field, complexity of networked systems is a major area for research. Although systems engineering has advanced significantly since its inception, there is still a gap as it fails to fully look external to the system and determining how it fits into the overall network of system that begins to emerge as independent systems begin to operate in the same time and space.

18.2.2 System of Systems and Family of Systems

As systems evolve, system engineers have expanded the methodologies to understand both systems of systems (SoS) and family of systems (FoS). Large, complex systems are increasingly composed of a variety of component systems that are newly developed, modified commercial-off-the-shelf (COTS) systems, and/or existing systems [17]. Similar to SoS, FoS are independent systems; however, they differ in that FoS operate in the same domain but may not provide additional emergent value.

A significant amount of literature focuses on identification and definition of SoS; however, the literature also identifies gaps in methodologies to analyze and design SoS. SoS are integrated systems comprised of individual systems that provide value by themselves, but when combined, they provide emergent behavior and greater value than the individual contributions. Keating, et al. described a SoS as "A metasystem, comprised of multiple embedded and interrelated autonomous complex subsystems that can be diverse in technology, context, operation, geography, and conceptual frame" [18]. Maier describes a SoS as an assemble of components, individual systems themselves, with the distinction that they may be managed and operated independently of the SoS [19]. Consistent with these descriptions of SoS, Boardman and Sauser propose five characteristics of SoS: autonomy, belonging, connectivity, diversity, and emergence [20]. In addition, INCOSE has defined SoS as "a system-of-interest whose system elements are themselves systems; typically these entail large scale inter-disciplinary problems with multiple, heterogeneous, distributed systems" [9].

There is not as much literature focused on FoS, and it is generally used in conjunction with SoS. Whereas in a SoS, the combination of a group of integrated subsystems provides greater value than the individual contributions; in a family of systems, the parts are not necessarily integrated, but they operate in the same domain to provide similar value to the stakeholder [21]. Often, the FoS construct is used to manage similar systems. The DoD defines FoS as "a set of systems that provide similar capabilities through different approaches to achieve similar or complementary effects" [22]. Again, the FoS construct does not necessarily consider the entire network of systems, but only the systems that have been grouped together as a FoS.

The literature for analysis and design of SoS presents an opportunity to develop methodologies and techniques to manage the complexities and scale of SoS. INCOSE identified several challenges to the design and analysis of SoS: system elements operate independently; elements have different lifecycles; complexity of system interaction, management overshadows engineering; and the fact that initial SoS requirements can be ambiguous [9]. Some efforts have proposed methodologies to design SoS, the SoS Tradespace Exploration Method (SoSTEM) assists decision makers in exploration of the tradespace during the conceptual design phase. SoSTEM provides a quantitative method to generate SoS attributes by combining component system attributes and examines the SoS's value deliver over time through multiepoch analysis for a large number of design alternatives [23]. Other work has focused on designing resiliency within SoS by focusing on improving nodes with the SoS or using modeling and simulation to understand the propagation of failures within a SoS [24]. Although SoS analysis is beginning to expand to incorporate additional methods, there is still room for additional research in this area.

18.2.3 System Architecture

System architects use architectures to understand, design, and manage complex systems and potentially develop standards across the discipline for these architectures [25]. System architecture is a method to design complex systems and understand how their function and combine to provide value for the stakeholder. As such, system architects attempt to understand and document how the system interacts with external forces to include other systems. Maier and Rechtin describe a system's architecture as a description of "whatever aspects of physical structure, behavior, cost, human organization, or other elements are needed to clarify the client's priorities" [26]. A good system architecture will simplify the presentation of the system's structure and assists decision makers in understanding the complex relationships within the system.

A system's architecture is composed of three major components: functional, what the system must do; physical, how the physical components perform the functions; and operational, how the functions consume resources [14]. Like systems engineers, system architects decompose a system into modules and allocate components to functions in a one-to-one mapping. However, much like systems engineering, system architectures begin with the system of interest and decompose the architecture into levels of details to understand the inner workings of a complex system. Architectures do specify interactions with elements outside the system boundary, but generally only one order away from the system of interest and do not capture the network of systems that emerges across an enterprise.

One effort to move beyond individual system architectures is examining patterns within system architectures, which shows how patterns found in one set of architectures may apply more broadly to other domains that have similar needs. Cloutier and Verma discuss how patterns found in one set of architectures may apply to other domains that have similar needs and developed a concept for understanding patters in system architecture [27]. In previous work, they identify and document a pattern for command and control for military systems that emerged from analysis of several individual architectures [28]. They go on to propose that once a pattern is identified in one domain, this case the military domain, they can apply the pattern to other areas, emergency response or city management, to reuse elements of the architecture to assist in the development of other systems. This work moves toward the concept of combining multiple architectures to gain benefit at the enterprise level but remains focused internally to a system or system of systems.

18.3 Methodology

This section presents the background for the proposed methodology to understand and manage the complex network of systems that has emerged within the DoD over the past several decades. It introduces social network analysis as a set of tools and methods that may assist systems engineers and architects in dealing with emergent networks of systems. In addition, it identifies potential social network analysis metrics that may prove to be useful in understanding emergent networks of systems.

18.3.1 Social Network Analysis

Social network analysis is a specific application of network analysis sociologists use to analyze social networks in which the nodes are individuals, groups, or organizations that are related together by various means. Sociologists construct networks through a variety of interdependencies to include shared interests, social contacts, membership in organizations, participation in events, family ties, or financial ties [29]. Social network analysis models represent a network's nodes (people) and edges (relationship) and allow for the calculation of mathematical metrics that provide valuable information about individual nodes (degree centrality, closeness, and betweeness) in the network as well as the network itself (centralization, density, and cohesive subgroups) [30]. One distinction of social network analysis is that it places emphasis on the relationships between the nodes as opposed to the attributes of individual nodes [31].

An important aspect of social network analysis is that multiple relationship types can be merged into one network to identify hidden relationships and provide insight into the overall structure of the network. White, Boorman, and Breiger discuss a method to determine social structure from multiple networks that independently do not uncover the true social structure, but when aggregated, they begin to display the true underlying network of people [32]. There are several challenges associated with social network analysis that align with challenges in networks of systems as well. Social networks are dynamic; they often have fuzzy boundaries that may not be immediately evident, and it is likely that the data on the network are not complete [33]. However, social network analysis has overcome some of these challenges through the use of network analysis algorithms and advanced computer process and visualization to incorporate hundreds of nodes and relationships.

18.3.2 Social Network Analysis' Individual Metrics

The first individual node metric from social network analysis that may have value to systems engineers is degree centrality. Degree centrality is based on that notion that actors with the most ties to other actors must be important within the network [34]. Graphically these would be nodes with the most edges emerging from them to other nodes within the network. Mathematically, in a network of g actors and individual nodes with d connections:

$$C'_d(n_i) = \frac{d(n_i)}{g-1} \tag{18.1}$$

This has been used by sociologists to identify actors that are very active in the network, which has motivated them to develop and use the degree centrality metric [35]. This enables them to identify nodes within the network that are well connected to the rest of the nodes in the network.

The degree centrality could provide an assessment of the interoperability of a system within a network of systems. Generally, a DoD system is considered interoperable if it can enter into the DoD's network, but this fails to consider or account for physical interoperability or resource flows. However, in a network of systems, the degree centrality metric could determine how connected the individual system is to the overall network and thus determine the true interoperability of a system.

Another measure of centrality is the closeness metric, which views an actor's centrality based on the distance from that node to all the other nodes in the network [34]. For this metric to be meaningful, the nodes in the network must be connected to the node of interest, otherwise the distance becomes infinite. Mathematically, closeness centrality, in the network described above, is calculated by:

$$C'_c(n_i) = \frac{g-1}{\sum_{j=1}^{g} d\left(n_i, n_j\right)} \tag{18.2}$$

This provides insights into the speed at which information or resources travel along a network between the different nodes. This can be used to identify actors that may receive information in a gossip network first or identify companies that may be first to develop and bring a product to market [36].

For systems engineers, closeness centrality could help them determine information flows across the network. When analyzing cyber networks, this could assist in the understanding of how vulnerable the entire network is to an attack. Likewise, in an infrastructure network, it could identify nodes that would quickly restore power to the entire grid. Within the DoD network of systems, it may assist analysts in their understanding of how interoperable a network of systems is or identify critical systems in the network.

The third individual node metric that may provide value to systems engineers in their analysis of networks of systems is the betweenness centrality. Also known as a bridge, this metric identifies situations in which interactions between two nonadjacent nodes in the network depend on another actor in the network [34]. Going back to the Bin Laden example, this was the courier who relayed messages between Bin Laden and other high members of al-Qaeda who's identification was the key to finding Bin Laden [1]. There are several assumptions that go into the calculation of betweenness, but it is based on the idea that communication in the network will follow any path and uses the probability of a communication for the calculation. Mathematically, betweenness is represented by the following equation, where g_{jk} is the number of edges linking j and k:

$$C_B(n_i) = \sum_{j<k} g_{jk}(n_i)/g_{jk} \tag{18.3}$$

Traditional social network analysis uses the betweenness metric to identify actors that have the potential to mediate flows of information between groups of actors [35]. In this manner, nodes with a high betweenness score act as bridges between subgroups of actors in the overall network.

For networks of systems, identifying bridges within the network could provide valuable information to systems engineers as they could identify systems that are critical for the operation of the overall network. These bridge systems may represent systems that must be hardened to develop a robust network of systems. In addition, these nodes in the network may require additional attention if they fail, as in an infrastructure network, as they will return service to the most number of customers. Within the DoD network of systems, they may require additional systems to ensure that the DoD maintains redundant systems for critical tasks.

18.3.3 Social Network Analysis' Network Metrics

This section focuses on metrics associated with the entire network that may provide valuable insights into the structure and behavior of the network as a whole. Generally, centrality is focused on individual nodes in the network; however, a group centralization metric can be calculated. The group centralization metric begins to identify how distributed the network is and the dispersion of connections within the network [34]. The general centralization metric is mathematically calculated as follows:

$$C_A = \frac{\sum_{j=1}^{g} \left[C_A(n^*) - C_A(n_i)\right]}{\max \sum_{j=1}^{g} \left[C_A(n^*) - C_A(n_i)\right]} \tag{18.4}$$

This can be used to determine if there are actors in the network that completely dominate the network when the metric is near 1, or if all actors are relatively equal when the metric is near 0 [34].

Systems engineers may use this to understand reliability or robustness of the entire network of systems. From a customer perspective, the customer may not care which node in the network is providing them with a service; however, if the entire network fails and does not meet their expectation, this could create a problem. Within the DoD, a warfighter may not care which system is providing them support, such as in a close air support role, but if that capability is not available, it could result in overall mission failure. It may also indicate when there is an unbalance within the network and the entire network of systems is heavily dependent upon one node in the network.

Similar to centralization, the density of the network identifies how many connections are present in the network when compared to the maximum number of potential connections based on the total number of nodes in the network [33]. In the same network as before, where g represents the number of nodes and d represents the individual nodes connections, this is represented mathematically as follows:

$$D = \frac{\sum_{i=1}^{g} d(n_i)}{\frac{g(g-1)}{2}} \tag{18.5}$$

In a terrorist network, the density may indicate how effective the network is at distributing information as there are several connections in the network, but it also may indicate that the network is prone to failures [37]. Generally, in a network, high density would be considered a good thing because most of the nodes are connected. However, Everton argues that in a dark network or terrorist network, a highly dense network may lead to failures as it is dependent upon several connections [37].

For emergent networks of systems, density may be an indication of robustness of the network if the network is constructed with systems and functions. This would indicate that multiple systems can perform the same function and continue to provide value to the stakeholders. In addition, as described by Everton, this may also indicate networks of systems that are highly reliant on each other, overly interdependent, that may be prone to overall network failure based on one or two of the key nodes failing. This was recently evident in the massive power outage in Puerto Rio, in which over 1.5 million customers were without power after a fire at a transfer station in the Puerto Rican power network [38].

Another network metric that may help understanding emergent networks of systems is cohesive subgroups. These are subsets of actors in a social network among whom there are strong, direct, intense, or positive relationships [34]. Graphically, these can be identified as tight clusters of nodes in a network, and mathematically, these are represented by *n*-cliques. An *n*-clique is a maximal subgraph, in which, the largest distance between any two nodes in the subgraph (N_s) is not greater than *n* [34]. Mathematically:

$$d(i,j) \leq n \text{ for all } n_i, n_j \in N_s \tag{18.6}$$

Sociologists have used this to identify small subgroups of people or the "small-world" network of closely connected individuals in a larger, more complex network [39]. In addition to identifying smaller cliques of individuals within the network, they can also identify larger peer groups that reside in a larger network.

Within the systems engineering realm, these cohesive subgroups may be used to identify systems that are behaving like system of systems. This metric could potentially identify groups of systems that have the characteristics of a system of system as described by Boardman and Sauser. Two of the main characteristics of a system of system are belonging: an individual system becomes part of the system of systems, and connectivity: that there are connections between the systems in a system of systems [20]. Through the use of this metric, systems engineers may be able to identify previously unknown system of systems or identify legacy systems in the network that should be managed as a system of systems.

18.4 Initial Results

The concept of an emergent network of systems builds upon systems architecture and network analysis by applying social network analysis tools and methods to better understand networks of systems. The individual systems in a network of systems are represented as nodes of the network and the various relationships from the individual system's architecture become the edges of the network. This could have a profound impact on the understanding of networks of systems, reliability of the network of systems, and the design of both the network and individual systems within the network. In addition, these tools and patterns within the network of systems may lead to the identification of potential systems of systems or families of systems as part of the design process. This section presents initial results of applying social network analysis to the DoD network of systems.

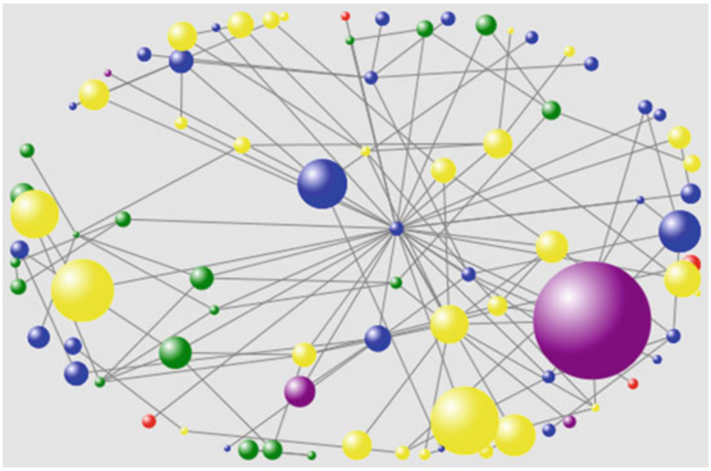


Fig. 18.1 DoD network of systems

18.4.1 DoD Network of Systems

The actual DoD network of systems is comprised of hundreds of legacy, underdevelopment, and planned systems that range in cost from several million to over a trillion dollars. In this initial work, the network of systems is based on the GAO's report on Major Defense Acquisitions Programs (MDAP). This network is comprised of 79 MDAPs, 27 planned MDAPs, and 8 legacy systems that are mentioned in the report [40]. Figure 18.1 presents a subset of the entire network of DoD systems as outlined in the GAO report along with the connections between systems that are described in the report. Within the figure, the size of the node represents the total acquisition cost of the system, and the colors represent the various services that are sponsoring the program.

As shown in Fig. 18.1, the graphical representation of the network along begins to provide insights for a systems engineering or DoD analyst. First, the node in the middle of the network is the GPS III system, which provides global positioning and timing for almost all DoD systems. The central position and the number of edges that extend from this node demonstrate the importance of this system within the entire DoD network. In addition, as the node size represents the total acquisition cost, the Joint Strike Fighter (JSF), the large purple node to the right, and other high-cost nodes within the network stand out.

Figure 18.2 presents a subset of the entire network, which focuses on the known littoral combat ship (LCS) family of systems. In the center of the four highlighted nodes is the LCS seaframe, which provides the hull and the basis for the entire family of systems. The three nodes that extend from this node are the various mission modules, which provide different capabilities to the warfighter. Through this analysis, cliques or subgroups may be able to identify actual or potential family

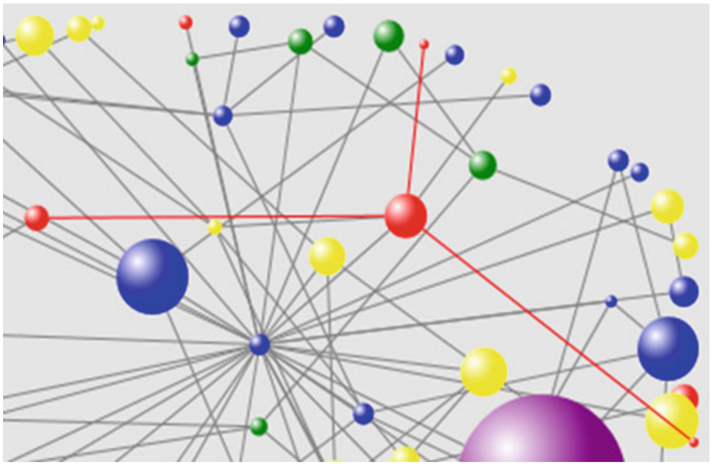


Fig. 18.2 Littoral combat ship family of systems

of systems that may help DoD systems engineers in the design and management of acquisition programs.

18.5 Conclusion and Future Work

This section presents a summary of the key findings of the work and the potential for future work to continue to explore the idea of applying social network analysis tools and metrics to emergent networks of systems. The major finding of this work is that there is potential to use a variety of social network analysis tools and techniques to networks of systems. Even just representing the network graphically provides initial visual insights into the network and the systems within the network. However, there is a large opportunity for further exploration of these methods to better understand emergent networks of systems, validate that social network analysis metrics can provide insights into the network, and expand beyond just DoD systems.

There are three major areas for future work, which could provide additional insights into the emergent network of systems analysis. First, a further application to DoD Systems along with a verification and validation of the model may provide systems engineers with a set of social network analysis metrics that are insightful. For the DoD, this may assist with the management of both planned, underdevelopment, and legacy systems. Additional analysis of cyber networks or infrastructure networks may provide additional validation of the applicability of social network analysis metrics to networks of systems.

Another are for future work is to examine the "ilities," potentially interoperability, robustness, reliability, and flexibility, and to determine if social network analysis metrics may provide quantifiable metrics for these "ilities." Finally, implications for systems design should be explored to determine if an analysis of the current network of systems may help systems engineers design new systems. This could lead to an analysis of both the current network of systems and the planned network based on the proposed systems architecture. This area could be further explored to determine if systems engineers could or should manage and design the entire network of systems.

References

1. Schmidle N (2011) Getting Bin Laden. The New Yorker, 8 August 2011
2. Owens WA (1996) The Emerging U.S. System-of-Systems, National Defense University Strategy Forum
3. Dahmann J,Baldwin K (2008) Understanding the Current State of US Defense Systems of Systems. In: Systems Conference, Montreal
4. Government Accountability Office (2015) Opportunities exist to improve the department of defense's portfolio management. GAO, Washington, D.C
5. Joint Chiefs of Staff (2012) Manual for the operation of the joint capabilities integration and development system. Department of Defense, Washington DC
6. Architecture Frameworks Working Group (2009) DoDAF satisfaction of systems engineering needs. National Defense Industrial Association, Washington, D.C.
7. Ring S, Nicholson D, Thilenius J, Harris S (2004) An activity-based methodology for development and analysis of integrated DoD architectures –"the art of architecture". MITRE, Bedford
8. Enos J (2014) "Synthesizing DoDAF Architectures to Develop the Joint Capability Enterprise Architecture," in Systems Engineering D.C, Washington, D.C
9. INCOSE (2007) Systems Engineering Handbook v3.1. International Council on Systems Engineering
10. Kossiakoff A, Sweet W, Seymour S, Biemer S (2011) Systems engineering principles and practice, 2nd edn. Wiley, Hoboken
11. Parnell G, Driscoll P, Henderson D (2011) Decision making in systems engineering and management, 2nd edn. Wiley, Hoboken
12. Goode H, Machol R (1957) Systems engineering: an introduction to the design of large-scale systems. McGraw-Hill, New York
13. INCOSE. About Incose (2015) [Online]. Available: http://www.incose.org/about. Accessed 1 May 2016
14. Buede D (2000) The engineering design of systems: models and methods. Wiley, New York
15. McManus H, Richards M, Ross A,Hastings D (2009) A Framework for Incorporating "ilities" in Tradespace Studies, in American Institute of Aeronautics and Astronautics
16. Rouse W (2007) Complex engineered, organizational and natural systems. Syst Eng 10:260
17. Sage A, Cuppan C (2001) On the systems engineering and management of systems. Syst Manag 2:325–345
18. Keating C, Rogers R, Unal R, Dryer D, Sousa-Poza A, Safford R, Peterson W, Rabadi G (2003) Systems of systems engineering. Eng Manag J 15:36–45
19. Maier MW (1998) Architecting principles for systems-of-systems. Syst Eng 1:267–284
20. Boardman J,Sauser B (2006) System of Systems – the meaning of of, in 2006 IEEE/SMC International Conference on System of Systems Engineering, Los Angeles
21. Clark J (2008) System of systems engineering and family of systems engineering from a standards perspective. In: IEEE international conference of systems of systems engineering

22. Director, systems and software engineering (2008) In: Systems engineering guide for systems of systems. Department of Defense, Washington, D.C.
23. Chattopadhyay D, Ross A, Rhodes D (2009) A practical method for tradespace exploration of systems of system. In: AIAA Space, vol 2009, Pasadena
24. Uday P, Marais K (2015) Designing resilient systems-of-systems: a survey of metrics, methods, and challenges. Syst Eng 18:491–510
25. Crawley E, de Weck O, Eppinger S, Magee C, Moses J, Seering W, Schindall J, Wallace D, Whitney D (2004) The influence of architecture in engineering systems, Cambridge
26. Maier MW, Rechtin E (2002) The art of systems architecting, 2nd edn. CRC Press, New York
27. Cloutier R, Verma D (2007) Applying the concept of patterns to systems architecture. Syst Eng 10:138–154
28. Cloutier R, Verma D (2006) Applying pattern concepts to enterprise architecture. J Enterp Archit:34–50
29. Serrat O (2010) Social network analysis. Asian Development Bank
30. Freeman L (2004) The development of social network analysis. Empirical Press, Vancouver
31. Hanneman R, Riddle M (2005) Introduction to social network methods. University of California, Riverside
32. White H, Boorman S, Breiger R (1976) Social structure from multiple networks. Am J Sociol 81:730–780
33. Everton S (2012) Disrupting dark networks. Cambridge University Press, New York
34. Wasserman S, Faust K (1994) Social network analysis: methods and applications. Cambridge University Press, New York
35. Faust K (1997) Centrality in affiliation networks. Soc Networks 19:157–191
36. Borgatti SP (2005) Centrality and network flow. Soc Netorks 27:55–71
37. Everton S (2009) Terrorist Networks. In: association for the study of economics, religion and culture, Washington, D.C
38. CNN. Nearly 1.5 million without power in Puerto Rico. CNN, 22 September 2016. [Online]. Available: www.cnn.com. Accessed 24 Sept 2016
39. Moody J (2001) Peer influence groups: identifying dense clusters in large networks. Soc Networks 23:261–283
40. Government Accountability Office (2016) Defense acquisitions: assessment of selected weapon programs, Washington, D.C

Part III
Tradespace Visualization and Exploration

Chapter 19
Designing for System Value Sustainment Using Interactive Epoch–Era Analysis: A Case Study from Commercial Offshore Ships

Michael D. Curry, Carl F. Rehn, Adam M. Ross, and Donna H. Rhodes

Abstract This paper applies the Interactive Epoch–Era Analysis (IEEA) framework on a case study from commercial offshore ship design, incorporating techniques from visual analytics such as interactive visualizations to gain insight from large, high-dimensional data sets, resulting in improved strategies for value sustainment. New prototype visualizations are described that are motivated by a need to address design questions that are not well-suited for analysis with metrics, often applied in other EEA case studies, such as fuzzy Pareto number (FPN) or fuzzy normalized Pareto tract (fNPT) alone. For the offshore ship design case, this includes assessing the trade-off between designs optimized to target the primary mission versus being robust for uncertain subsequent missions. Further, considerations related to the implementation of interactive visualization applications, such as scalability and latency, are discussed emphasizing a need for continued research on methods for effectively handling large, high-dimensional data sets in design of complex systems under uncertainty.

Keywords Systems engineering • Tradespace exploration • Resilience • Epoch–era analysis • Interactive • Visualization • Visual analytics • Ship design

19.1 Introduction

Epoch–Era Analysis (EEA) is designed to clarify the effects of changing contexts over time on the perceived value of a system in a structured way [1, 2]. Prior research studies have demonstrated the usefulness of Epoch–Era Analysis (EEA), but some challenges remain in practical applications for informing decision-making in many real-world problems. Although methods for implementing EEA constructs

M.D. Curry (✉) • A.M. Ross • D.H. Rhodes
Massachusetts Institute of Technology, Cambridge, MA, USA
e-mail: curry@mit.edu

C.F. Rehn
Norwegian University of Science and Technology, Trondheim, Norway

have been developed and applied in case studies, a prescriptive framework that explicitly considers human interaction does not yet exist. Furthermore, EEA can result in large, multivariate data sets that are difficult to manage, visualize, and analyze. This is precisely where effective visualization and analysis techniques are needed, to help make informed decisions, design successful strategies for value sustainment, and derive valuable insights from data.

Interactive Epoch–Era Analysis (IEEA) is an iterative framework for concept exploration that provides a means of applying EEA constructs while controlling growth in data scale and dimensionality [3, 4]. IEEA leverages interactive visualization, which prior visual analytics research has demonstrated to be a valuable tool for helping analysts gain insight from underlying data in exploratory analyses, like early-phase concept selection, resulting in improved decision-making. Consequently, the extension of interactive visualization to system design problems with life cycle uncertainty can result in improved comprehension of the nature of underlying trade-offs and can also simultaneously improve a designer's ability to communicate their decision-making rationale to others.

19.2 Methods: Framework for Interactive Epoch–Era Analysis (IEEA)

IEEA leverages human-in-the-loop (HIL) interaction to manage challenges associated with the large amounts of data, potentially enabling structured evaluation and visualization of many design alternatives across many different futures and potential life cycle paths. This new approach enables the design of systems that can deliver sustained value under uncertainty.

19.2.1 Extension of Prior EEA-Based Methods

Prior research on methods and processes for applying EEA form the basis for the framework described in this paper. The Responsive Systems Comparison (RSC) method, proposed by Ross et al. [5, 6] as a prescriptive method for applying multi-attribute tradespace exploration (MATE) and EEA, was developed to study system value sustainment through changeability. More recently, Schaffner [7] proposed the RSC-based Method for Affordable Concept Selection (RMACS) that expands the original seven processes of RSC to nine; it explores the application of multi-attribute expense (MAE) to more effectively capture resources and expenditures required to realize a given system.

However, IEEA differs from both RSC and RMACS in that it strongly emphasizes iteration and human-in-the-loop (HIL) interaction throughout the process. Iteration is necessary because the analysis is inherently exploratory in nature.

HIL interaction is necessary because the problem is multifaceted and complex, not strictly deterministic. It is also not necessarily intended as a unique or exact prediction of system performance or of future events, and thus needs a decision-maker's intervention moving forward in the analysis. As is quite often the case, there is both uncertainty and potential for errors in assumptions and model implementation, where human judgment is necessary to provide context and make sense of the data. Therefore, by its nature, this is not a problem that can be handed over completely to an automated optimization algorithm though some level of automated analysis could be beneficial to the decision-maker.

19.2.2 Description of IEEA Framework Modules

The purpose of IEEA, much like the purpose of RSC as described by Ross et al. [5], is to "guide the...practitioner through the steps of determining how a system will deliver value; brainstorming solution concepts, identifying variances in contexts and needs (epochs) that may alter the perceived value delivered by the system concepts; evaluating key system trade-offs across varying epochs (eras) to be encountered by the system, and, lastly, developing strategies for how a designer might develop and transition a particular system concept through and in response to these varying epochs." To that end, as shown in Fig. 19.1, the IEEA framework is characterized by ten individual processes that can be abstracted into six main modules:

1. *Elicitation* of relevant epoch and design variables (often through interviews)
2. *Generation* of all epochs, eras, and design tradespaces (often including enumeration)
3. *Sampling* of epochs and eras in which to evaluate design choices
4. *Evaluation* of designs in sampled subset of epochs and eras
5. *Analyses* of design choices in the previously evaluated epochs and eras
6. *Decisions* of final designs based on iterative evidence from previous modules.

While the sequence of these modules flows logically, IEEA is intended to be an iterative process, where users can go back and change responses within earlier modules at any point to reflect what they have learned from later ones. The six modules are composed of the ten processes within, but, depending on the nature of the study and the type and fidelity of information available to the analyst, it is not strictly required that each process step is applied.

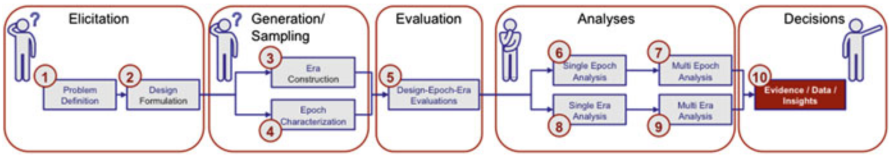


Fig. 19.1 Interactive epoch–era analysis process and modules

19.2.3 Implementation of a Visual Analytics Application for IEEA

Visual analytics applications in other domains have shown promise for solving problems whose size, complexity, and need for closely coupled human and machine analysis may make them otherwise intractable [8, 9]. A primary focus of research on the IEEA framework has been toward the development of visual analytics applications that can be employed to evaluate and demonstrate how enhanced human interaction techniques and visualizations can aid in the analyses and sense-making of high-dimensional multivariate EEA data sets.

The exact visualization and interaction approaches for implementing each of the process steps are nonunique and various issues can arise, depending on the specific way in which a visual analytics application is implemented. For example, the specific software implementation of an interactive visualization application requires trade-offs in development effort, extensibility, comprehensibility, scalability, and interaction latency. The exact visualization or analysis that is most useful to an analyst applying the IEEA framework is largely case-dependent. Therefore, the prototype visualizations developed for this research are not intended as a one-size-fits-all application but, rather, as examples that demonstrate techniques and interactive visualization types that can be customized for a specific case.

Many of the techniques discussed in Curry et al. [3, 4] can be applied to facilitate a practical implementation of IEEA. For example, online analytical processing (OLAP) techniques can be applied to improve data handling, which enables scalability to larger data sets with more designs, epochs, or other data dimensions. Interaction latency is a common issue as the size of the data set grows. Prior research has shown that increasing latency can adversely affect user performance in exploratory data analysis [10]. This is largely task-dependent, however, so a user-centric approach is recommended rather than focusing on reducing latency across all analysis stages. For operations that are sensitive to latency, such as brushing and linking between coordinated views, methods such as caching, precomputing, and prefetching can be applied to minimize latency. Trade-offs must be considered though, because it is not always beneficial or possible to anticipate or precompute every conceivable piece of data that the user might be interested in evaluating.

19.3 Commercial Offshore Ship Case Study

This case study is based on the one described by Rehn et al. [11]. A more detailed discussion of the case setup is provided in that paper.

Offshore ships, in contrast to traditional deep-sea cargo ships, are designed to provide special operational services typically related to the offshore oil and gas industry. This group of ships comprises platform supply vessels (PSV), inspection maintenance and repair (IMR), and offshore construction vessels (OCV), to mention a few. A recent period of high oil prices and deep-sea petroleum discoveries has spurred the development of offshore oil and gas fields. Thus, there has been a growing need for offshore services, including well maintenance and intervention services with light, riserless technologies. OCVs have taken an increasingly large part in the development of these, in particular for the marginal fields, due to their price competitiveness. Additionally, the Deepwater Horizon oil spill in 2010 in the Gulf of Mexico has changed some of the focus for the offshore shipowners toward being able to provide various deepwater emergency and rescue operations. This strong market period has characteristically driven the design of offshore ships toward multifunctional, *gold-plated*, and expensive solutions [12]. However, the recent oil price collapse of 2014 has had a significant impact on the offshore markets, rendering many of these multifunctional ships less competitive against cheaper, specialized ships. The current situation in the offshore industry serves as a good example of the importance of focusing on value robustness and operational flexibility as key factors for success in a highly volatile maritime industry [13, 14].

Offshore ships are usually built either for a specific long-term contract or on speculation. A long-term contract may last 5–10 years, and these ships are often specialized for the particular mission. Ships built on speculation tend to be more multifunctional, to be able to take on different contracts. If these ships do not get any lucrative long-term contracts, they are often offered in the spot market to take on various short-term contracts. If a ship does not get a contract, it is idle for short periods or laid up over longer periods. This case study motivates several questions, the evaluation of which may be aided using interactive applications described in this paper and by prior IEEA case studies:

1. What is the trade-off between optimizing for the primary contract and making the design robust to more than one contract in terms of the number of acceptable designs in the tradespace?
2. What is the impact in terms of both cost and reduced performance when attempting to ensure that designs satisfy all potential contracts?
3. What are the benefits and drawbacks of active versus passive value robustness?
4. Which contracts are most challenging to satisfy?

19.3.1 Process 1: Value-Driving Context Definition

The first process defines the stakeholders, problem statement, exogenous uncertainties, and the basic value proposition for the system. The business opportunity for a new offshore ship design emerges from an expected strong demand for offshore oil and gas over the next couple of decades, despite recent short-term oil price volatility. The Deepwater Horizon accident has further resulted in an increased focus on being able to provide advanced offshore emergency services in the Gulf of Mexico. An offshore shipowner wants to target this business opportunity, and, in particular, a potential five-year contract for a large oil company. The shipowner wants to have a profitable and eco-friendly solution.

19.3.2 Process 2: Value-Driven Design Formulation

The second process begins by defining the statements of needs, which become the attributes of system performance, along with utility functions describing the preference for each attribute. The system boundary for the single ship design is around the ship itself and does not consider, for example, the total profitability of the overall shipping company. Profitability is a measure of the ability of the design to generate profits, and eco-friendliness represents the ability of a design to reduce emissions during operation and transit. The nonmonetary and monetary value attributes are kept separate due to their temporal differences in the model, which is further discussed in Rehn et al. [11]. In the model, profitability is considered at the era level, while eco-friendliness is considered at the epoch level.

Even though *value-focused thinking* involves exploring various high-level solution forms, we assume the form of a standard single-hull OCV for demonstration purposes in this case study. The following ship-level design variables are considered: length, beam, depth, power, accommodation, main crane, light well intervention tower, moonpool, fuel type, dynamic positioning, remotely operated vehicle (ROV), pipe laying capability, and design for changeability level.

19.3.3 Process 3: Epoch Characterization

In process 3, the key contextual uncertainties are identified so that epoch variables can be characterized. Based on the system boundary defined, eight epoch variables are identified, as illustrated in Fig. 19.2. These epoch variables represent mainly the details of missions for a ship, operationalized through the contract rate and technical requirements, and the details of the operational area including the sea state and water depth.

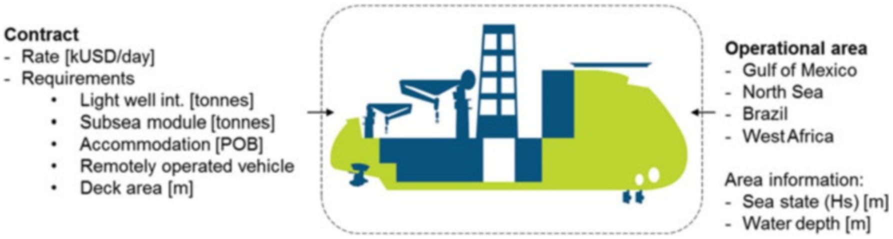


Fig. 19.2 System boundaries and epoch variables [11]

19.3.4 Process 4: Era Construction

This process constructs era timelines composed of multiple sequences of epochs each with a set duration to create long-run descriptions of possible future scenarios a system may encounter. Simulating life cycle performance in this way allows an analyst to evaluate path-dependent effects that may only arise when uncertainty is time-ordered. The activities in this process are in many ways analogous to those used in narrative or computational scenario planning. The future timelines can be constructed manually with the aid of expert opinion (narrative) or by implementing probabilistic models (computational), such as Monte Carlo simulation or Markov chain models that define epoch transitions.

Three narrative scenarios are considered in this case study. In two of the eras, the ship gets the targeted five-year contract initially, and experiences a relatively strong market the rest of the assumed 20-year lifetime. In the third era, the ship does not get the targeted contract due to a market collapse.

19.3.5 Process 5: Design–Epoch–Era Evaluation

The first four processes defined the relevant elements of the models that will be evaluated in the fifth process. The previously defined models are integrated to map design and epoch variables into stakeholder value and expense. Many techniques are available for assessing both value and expense of a given system. A generalized approach may include multi-attribute utility (MAU) to quantify stakeholder value and multi-attribute expense (MAE) to quantify expense. This step connects the value space and the design space in a mapping correspondence, often via an intermediate performance space. For the offshore ship, the various key performance indicators are estimated based on simple relations from the design variables, including speed, deck area, dead weight, acquisition cost, and operational costs. Designs that violate the technical requirements in an epoch are rendered infeasible.

19.3.6 Process 6: Single-Epoch Analysis

Single-epoch analysis is comparable to what is often referred to in practice as tradespace exploration. Within a given epoch, a scatter plot of cost (MAE) versus benefit (MAU) can be constructed that is fixed for short-run periods of stable context and needs (i.e., an epoch). Typically, decision-makers want to identify the frontier of Pareto optimal designs or, more generally, designs that are "close enough" to the Pareto front. Here the notion of "close enough" is operationalized through the Fuzzy Pareto Number (FPN) [15], which is used to quantify the distance from the Pareto Front for each design in each epoch. FPN is a "within-epoch" metric and its value for a given design will change in different epochs. Decision-makers can gain insights regarding the difficulty of a particular set of context and needs by visualizing how points move in the design space as the epoch and FPN values change. Additional insights may be gained from interactively filtering the design, performance, or value variables. This can be performed with the aid of the filtering application shown in Fig. 19.3, which allows the decision-makers to interact with their data to identify designs and epochs of interest. It also allows them to assign any of the defined variables to the radius, color, or x-y location of the points in the scatter plot to explore the data in four dimensions and better comprehend the behavior of the designs.

Figure 19.3 illustrates the tradespace for the offshore ship design base case, that is, the targeted contract with no technical requirements. Hence, at this initial stage, one can focus on understanding the dynamics of the underlying system. In this particular case study, the MAU only comprises one utility function, that is, eco-friendliness, even though the figure indicates a multi-attribute utility function on a general basis. The interactive filtering can aid in visualizing the exploration process of understanding the relative significance of individual design variables, as illustrated. For instance, filtering by beam and length, one can see that relatively slender ships tend to contribute to low FPN values. However, this again makes a design less stable in the water, which restricts the possibilities of retrofitting heavy equipment on deck without intervening with the main hull. Further, one can directly see the trade-offs of adding DFC levels, as design points shift right in the tradespace with increasing DFC due to increased cost.

19.3.7 Process 7: Multi-epoch Analysis

The activities of process 7 allow decision-makers to gain deeper insights by evaluating metrics between and across epochs to gauge the impact of uncertainties on system value. This includes the evaluation of short-run passive and active strategies for achieving value sustainment such that systems can maintain value delivery across different missions or changing contexts. A system that is passively robust is insensitive to changing conditions and continues to deliver acceptable

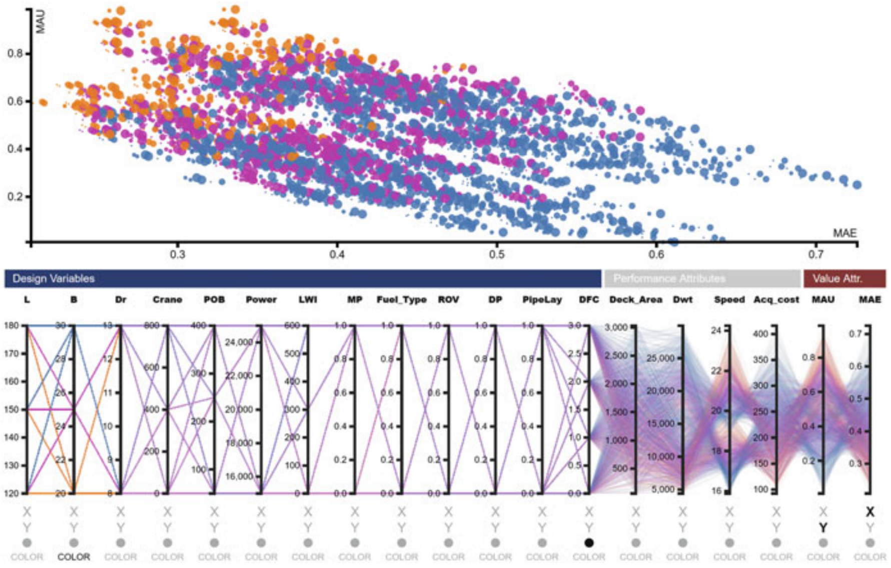


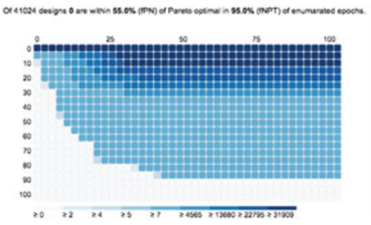
Fig. 19.3 Interactive filtering application for tradespace exploration for the offshore ship design base case

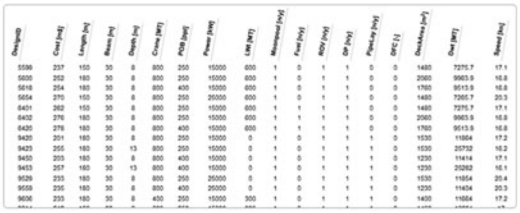
value. Alternatively, a system that suffers deterioration in value due to evolving conditions may benefit from the use of change options that make it flexible, adaptable, or resilient.

19.3.7.1 Evaluating Passive Strategies for Value Sustainment (Robustness)

In general, we want to have the lowest cost ship that can fulfill the technical requirements of a contract. Since ships that do not have the required technical equipment for an epoch are considered infeasible, the number of designs in the tradespace as well as its shape will change depending on the epochs. Equipment is typically a large cost driver; hence, trade-offs are likely required between optimality in any one epoch versus how many of the enumerated epochs can be satisfied when using passive strategies only. The percentage of enumerated epochs satisfied at a given fuzziness level is quantified using the fuzzy normalized Pareto trace (fNPT) metric. A proper exploration of the trade-off between “closeness” to the Pareto front (FPN) and passive robustness across various epochs (fNPT) is important when extracting insights from these large, high-dimensional data sets that are produced in the design process.

When examining this trade-off, attempting to look at all data dimensions of all possible designs across all possible epochs can be daunting for decision-makers. Even with clever visual encoding, visualizations that show all the data could likely





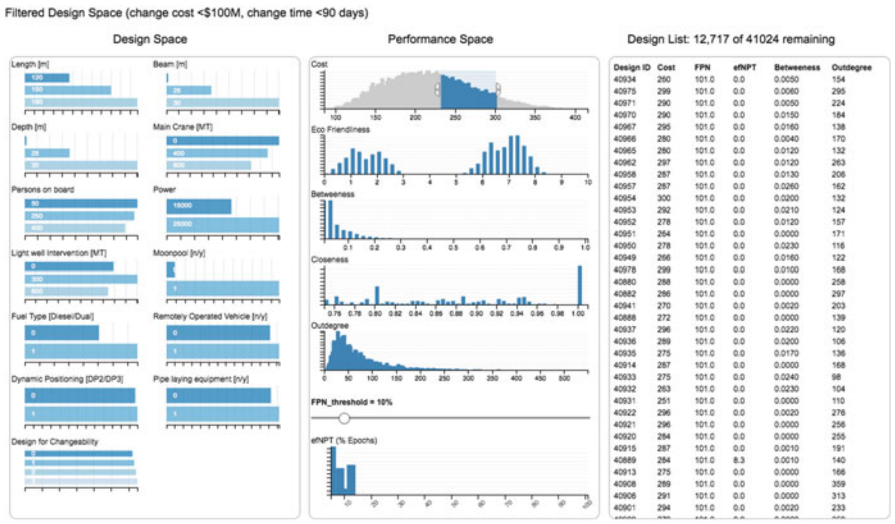
DesignID	Cost [m$]	Length [m]	Beam [m]	Depth [m]	Crane [MT]	POB [ppl]	Power [kW]	LWI [MT]	Moonpool [n/y]	Fuel [n/y]	ROV [n/y]	DP [n/y]	PipeLay [n/y]	DFC [-]	DeckArea [m²]	Dwt [MT]	Speed [kn]
5599	237	150	30	8	800	250	15000	600	1	0	1	1	0	0	1480	7275.7	17.1
5600	252	180	30	8	800	250	15000	600	1	0	1	1	0	0	2080	9903.9	16.8
5618	254	180	30	8	800	400	15000	600	1	0	1	1	0	0	1780	9513.9	16.8
5654	270	150	30	8	800	250	25000	600	1	0	1	1	0	0	1480	7365.7	20.3
6401	262	150	30	8	800	250	15000	600	1	1	1	1	0	0	1480	7275.7	17.1
6402	276	180	30	8	800	250	15000	600	1	1	1	1	0	0	2080	9903.9	16.8
6420	278	180	30	8	800	400	15000	600	1	1	1	1	0	0	1780	9513.9	16.8
9420	201	180	30	8	800	250	15000	0	1	0	1	1	1	0	1530	11664	17.2
9423	255	180	30	13	800	250	15000	0	1	0	1	1	1	0	1530	25732	16.2
9450	203	180	30	8	800	400	15000	0	1	0	1	1	1	0	1230	11414	17.1
9453	257	180	30	13	800	400	15000	0	1	0	1	1	1	0	1230	25282	16.1
9529	233	180	30	8	800	250	25000	0	1	0	1	1	1	0	1530	11854	20.4
9559	235	180	30	8	800	400	25000	0	1	0	1	1	1	0	1230	11404	20.3
9606	233	180	30	8	400	250	15000	300	1	0	1	1	1	0	1400	11664	17.2

Fig. 19.4 Interactive heatmap visualization (*left*), inspection table (*right*) for the ships in the selected tile in the heatmap

incur additional cognitive load for the users rather than reduce it. Fortunately, depending on the task they are focused on, an internal mental representation of all data is not strictly necessary for an analyst. The interactive heatmap visualization shown in Fig. 19.4 is one example of a simplified visualization that can show the compromise between Pareto efficiency (FPN) of designs within an epoch and the frequency with which they maintain that level of efficiency across multiple epochs (fNPT). This is illustrated in Fig. 19.4 for the offshore case and we can see that there are no designs that are Pareto optimal in all enumerated epochs. Accepting designs slightly away from the Pareto front or relaxing the constraint that epochs must be satisfied, allows additional design candidates to be identified. Figure 19.4 shows that the fuzziness (e.g., threshold FPN value) needs to be relaxed to approximately 45% for any designs to be in the fuzzy Pareto set for an estimated maximum 87% of all epochs. This indicates that, in fact, no ship can satisfy all contracts and that the most multifunctional passive ship can satisfy a maximum of 87% of the potential contracts requirements.

The interactive heatmap provides a high-level overview of trades between efficiency and robustness. But it does not answer the questions: if an analyst wants to examine more complex trade-offs, for example, how restrictions on cost or other performance attributes impact the trade-off between FPN and fNPT, or, alternatively, if an analyst wants to identify whether certain epochs, stakeholders, or context variables are more problematic than others for system value sustainment or they have a disproportionate effect on restricting the space of available alternatives. This type of information cannot be obtained from the heatmap visualization or aggregate measures like fNPT. More complex or nuanced questions like these require the examination of additional data dimensions that can be difficult to visualize and can also present added computational challenges.

This type of analysis is possible, however, with the aid of a more sophisticated visual interface like the example shown in Fig. 19.5, where a combination of online analytical processing (OLAP) and binned aggregation for fast filtering and interaction with larger data sets are applied. This visualization can also be easily scaled to case studies involving millions of designs and large numbers of data dimensions. This is possible because, rather than plotting every data point, each dimension is binned into a histogram that allows filters to be placed on individual data



Design List: 12,717 of 41024 remaining

Design ID	Cost	FPN	efNPT	Betweeness	Outdegree
40934	260	101.0	0.0	0.0050	154
40975	299	101.0	0.0	0.0060	295
40971	290	101.0	0.0	0.0050	224
40970	290	101.0	0.0	0.0150	184
40967	295	101.0	0.0	0.0160	138
40966	280	101.0	0.0	0.0040	170
40965	280	101.0	0.0	0.0120	132
40962	297	101.0	0.0	0.0120	263
40958	287	101.0	0.0	0.0130	206
40957	287	101.0	0.0	0.0260	162
40954	300	101.0	0.0	0.0200	132
40953	292	101.0	0.0	0.0210	124
40952	278	101.0	0.0	0.0120	157
40951	264	101.0	0.0	0.0000	171
40950	278	101.0	0.0	0.0230	116
40949	266	101.0	0.0	0.0160	122
40978	299	101.0	0.0	0.0100	168
40880	288	101.0	0.0	0.0000	258
40882	286	101.0	0.0	0.0000	297
40941	270	101.0	0.0	0.0020	203
40888	272	101.0	0.0	0.0000	139
40937	296	101.0	0.0	0.0220	120
40936	289	101.0	0.0	0.0200	106
40935	275	101.0	0.0	0.0170	136
40914	287	101.0	0.0	0.0000	168
40933	275	101.0	0.0	0.0240	98
40932	263	101.0	0.0	0.0230	104
40931	251	101.0	0.0	0.0000	110
40922	296	101.0	0.0	0.0020	276
40921	296	101.0	0.0	0.0000	256
40920	284	101.0	0.0	0.0000	255
40915	287	101.0	0.0	0.0010	191
40889	284	101.0	8.3	0.0010	140
40913	275	101.0	0.0	0.0000	166
40908	289	101.0	0.0	0.0000	359
40906	291	101.0	0.0	0.0000	313
40901	294	101.0	0.0	0.0020	233

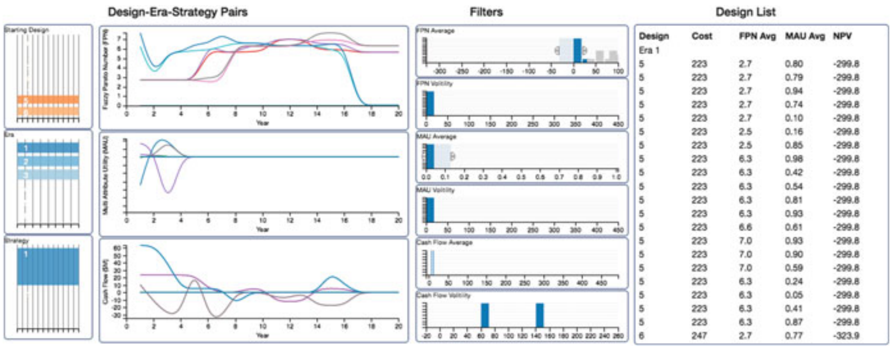
Fig. 19.5 Interactive filtering application implementing OLAP for the offshore case (Filtered for allowable change cost <$100 M and change time <90 days)

dimensions to see how that impacts the other dimensions in coordinated views. A list of candidate designs that match the filters is then displayed in the list on the right.

An analyst using this type of interactive visual interface can extract deeper insights about trade-offs by setting filters on various data dimensions to explore how those constraints impact other data dimensions or the list of available designs. For the commercial ship case study, this can be applied to gain a better understanding of the impact of fuzziness (FPN) and cost constraints. For instance, the designs that tend to be acceptable in most epochs, explored in the heatmap visualization in Fig. 19.4, also tend to be among the most expensive in the tradespace. In fact, no matter how much the fuzziness threshold is relaxed, there are no designs that have more than 85% of the epochs for a cost lower than $285 million. An analyst interested in achieving a lower target cost would need to examine in detail the cost savings that could be achieved by eliminating certain epochs (e.g., contracts, missions), which would result in a lower fNPT.

19.3.8 Process 8 and 9: Single and Multi-era Analysis

Implementation of changeability in the offshore ship case enables the system to mitigate risk and take advantage of opportunities in a future operational context. This is enabled by initially optimizing for the targeted contract, but also providing the flexibility to be able to change the design later based on the next state of operation, which is uncertain at the initial design stage.



Design	Cost	FPN Avg	MAU Avg	NPV
Era 1				
5	223	2.7	0.80	-299.8
5	223	2.7	0.79	-299.8
5	223	2.7	0.94	-299.8
5	223	2.7	0.74	-299.8
5	223	2.7	0.10	-299.8
5	223	2.5	0.16	-299.8
5	223	2.5	0.85	-299.8
5	223	6.3	0.98	-299.8
5	223	6.3	0.42	-299.8
5	223	6.3	0.54	-299.8
5	223	6.3	0.81	-299.8
5	223	6.3	0.93	-299.8
5	223	6.6	0.61	-299.8
5	223	7.0	0.93	-299.8
5	223	7.0	0.90	-299.8
5	223	7.0	0.59	-299.8
5	223	6.3	0.24	-299.8
5	223	6.3	0.05	-299.8
5	223	6.3	0.41	-299.8
5	223	6.3	0.87	-299.8
6	247	2.7	0.77	-323.9

Fig. 19.6 Interactive multi-era application implementing OLAP for the offshore case

An offshore ship may be seen as a movable flexible platform that can carry equipment that enables the ship to take on contracts of various types. The size of the platform may also change through, for example, elongation, but at a higher cost and duration, compared to more traditional equipment retrofits on deck. Single and multi-era analyses using interactive visualizations, as shown in Fig. 19.6, can aid in the assessment of different classes of changeability for the offshore design case. For brevity, this analysis is not discussed in this paper, but the interested reader is referred to previous demonstrations in prior case studies using IEEA [4] for further details.

19.3.9 Summary of Interactive Epoch-Era Analysis

The analyses outlined in the preceding sections provide a way for decision-makers to interactively evaluate the performance of multiple design alternatives across multiple futures. This creates opportunities for new insights at the expense of a potentially large and complex data set that can be difficult to analyze. The application of an interactive framework allows the users to visualize and engage with the data in new ways that can facilitate improved comprehension and decision-making. The insights that are extracted from this approach allow decision-makers to understand the characteristics of designs that can sustain value in all possible futures, through passive robustness or active changeability.

19.4 Conclusion

The research presented here applies the Interactive Epoch-Era Analysis (IEEA) framework, which provides a means for analyzing life cycle uncertainty when designing systems for sustained value delivery. Application of IEEA to a case study for commercial offshore ship design demonstrates key concepts and prototype interactive visualizations. IEEA extends existing frameworks with new analytic and interactive techniques that enable new capabilities and insights to be derived, which can lead to improved dynamic strategies for sustainment of system value delivery. In addition, these extensions enable the framing and analysis of large-scale design problems with uncertainty, such as the case study presented in this paper. Future work will extend this case study to include a deeper analysis of options at the epoch-level for changeability as well as era-level analysis of time-dependent aspects of system value.

Acknowledgments The authors gratefully acknowledge partial funding for this research provided through the Charles Stark Draper Fellowship program. This material is based upon work supported, in part, by the US Department of Defense through the Systems Engineering Research Center (SERC) under Contract HQ0034-13-D-0004. SERC is a federally funded University Affiliated Research Center managed by Stevens Institute of Technology. Any opinions, findings, and conclusions or recommendations expressed in this paper are those of the author(s) and do not necessarily reflect the views of the United States Department of Defense. The authors further gratefully acknowledge partial funding from the Norwegian Research Council and The Norway-America Association. Further, the authors acknowledge the prior contributions on the offshore ship design case study from Sigurd S. Pettersen, Jose J. A. Garcia, Prof. Stein O. Erikstad, Dr. Per O. Brett, and Prof. Bjørn E. Asbjørnslett.

References

1. Ross A (2006) Managing unarticulated value: changeability in multi-attribute tradespace exploration. PhD in Engineering Systems, Cambridge, MA
2. Ross A, Rhodes D (2008) Using natural. In: INCOSE International Symposium, Utrecht, Netherlands, June
3. Curry M, Ross A (2015) Considerations for an extended framework for interactive epoch-era analysis. In: 13th Conference on Systems Engineering Research, Hoboken
4. Curry M, Ross A (2016) Designing for system value sustainment using interactive epoch era analysis: a case study for on-orbit servicing vehicles. Systems Engineering
5. Ross A, McManus H, Long A, Richards M, Rhodes D, Hastings D (2008) Responsive systems comparison method: case study in assessing future designs in the presence of change. In: AIAA Space, San Diego
6. Ross A, McManus H, Rhodes D, Hastings D, Long A (2009) Responsive systems comparison method: dynamic insights into designing a satellite radar system. In: AIAA Space 2009, Pasadena
7. Schaffner M (2014) Designing systems for many possible futures: the RSC-based Method for Affordable Concept Selection (RMACS), with multi-era analysis, Cambridge, MA
8. Kosara R (2007) Visualization criticism – The missing link between information visualization and art. In: Proceedings of the 11th International Conference on Information Visualization (IV), pp 631–636
9. Chang R, Ghoniem M, Kosara R (2007) WireVis: visualization of categorical, time-varying financial transaction data. In: IEEE Symposium on Visual Analytics Science and Technology (VAST), pp 155–162
10. Liu Z, Heer J (2014) The effects of interactive latency on exploratory visual analysis. In: Comp. Graphics (Proc. InfoVis)

11. Rehn CF, Pettersen SS, Garcia JJ, Erikstad SO, Brett PO, Asbjørnslett BE, Ross AM, Rhodes DH (2016) Ill-structured commercial system design problems: the responsive system comparison framework on an offshore vessel case, Working Paper, Ed. Cambridge, MA, USA: MIT
12. Garcia JJ, Brandt UB, Brett PO (2016) Unintentieonal consequences of the golden era of the Offshore Oil & Gas industry. International Conference on Ships and Offshore Structures, September
13. Erikstad SO, Rehn CF (2015) Handling uncertainty in marine systems design – state-of- the-art and need for research. In: IMDC
14. Garcia JJ, Pettersen SS, Rehn CF, Ebrahimi A (2016) Handling commercial, operational and technical uncertainty in early stage offshore ship design.In: Conference on System of Systems Engineering, Kongsberg, Norway
15. Smaling R, de Weck O (2004) Fuzzy Pareto Frontiers in multidisciplinary system architecture analysis. In: 10th AIAA/ISSMO multidisciplinary analysis and optimization conference, Albany

Chapter 20
Simulation-Based Air Mission Evaluation with Bayesian Threat Assessment for Opposing Forces

André N. Costa and Paulo C.G. Costa

Abstract Several advancements have been made to the air mission planning process in recent years, spawning several software tools that allow for a quick analysis of the feasibility of the mission. However, the determination of the most likely outcomes of an air mission plan is still a challenge for modelers and planners. Mathematical models are capable of representing the complexity of sensors and weapons systems, but are not as effective in providing a thorough visualization of the mission, as well as in accounting for environmental factors and interactions between multiple systems within an operational setting.

Our research addresses the Systems Engineering problem of enhancing a system's response by predicting the adversarial reactions to different inputs. We adopt a physics-grounded simulation approach that focuses on the handling of uncertainty in behavioral models as a means of properly assessing the responses. As part of this research, the work presented in this paper proposes the use of high-resolution simulation for evaluating the survivability and mission accomplishment rates of an attacking aircraft during an incursion. The work differs from purely scripted simulation, which employs rule-based entities, predefined routes and behaviors for blue and red forces. Instead, our prototype relies on a Bayesian threat assessment and response methodology. Our goal is to represent the enemy's decision process, while being able to predict the mission outcomes and to correctly assess the regions of higher vulnerability.

Keywords Military simulation • Bayesian networks • Threat assessment • Survivability • Air warfare • Decision support

A.N. Costa (✉)
Institute for Advanced Studies, Brazilian Air Force, São José dos Campos, SP, Brazil
e-mail: negraoanc@fab.mil.br

P.C.G. Costa
Department of Systems Engineering and Operations Research, George Mason University, Fairfax, VA, USA
e-mail: pcosta@gmu.edu

A.M. Madni et al. (eds.), *Disciplinary Convergence in Systems Engineering Research*, https://doi.org/10.1007/978-3-319-62217-0_20

20.1 Introduction

When planning a mission, pilots need to handle multiple interrelated objectives, such as accomplishing the mission task, avoiding enemy fire and detection, and maintaining flight safety [1]. The latter is usually addressed by the mission planner software itself, which already takes into account information regarding aircraft fuel consumption, airspace restrictions, terrain collision avoidance, among many others. The remaining objectives, however, require a more sophisticated method for evaluation, since they are closely related to how capable the enemy's air defenses are and to how they operate.

However, information about the opposing air defense system is fraught with uncertainty, since the enemy would always attempt to negate or deceive its available capabilities or the location of its defense assets. That offers a challenge for planners and modelers on properly representing this uncertainty so as to achieve reliable results. Nevertheless, even if somewhat restricted, the available intelligence is one's best information, especially considering that it is the result of a thorough process of data collection and evaluation. In other words, this information is a major asset in estimating the opposing systems' positions and, therefore, their weapons and sensors ranges, based on prior knowledge on these systems' parameters obtained through intelligence and data collection activities.

The evaluation of the sensor system capabilities already computes other types of uncertainty. For instance, it relies on mathematical models that take into account factors such as the aircraft's radar cross section (RCS) and its exposure time to a given sensor [2]. This process includes not only characteristics of the observed unit, but also parameters of the sensor, such as, in the case of a radar, tracking algorithms, detection thresholds, signal processing methods, and antenna, transmitter, receiver, and exciter characterizations [3], all having inherent stochasticity in its modeling process.

All these factors explain the complexity of the process of merging information accruing from diverse sensors, weapon systems, and other input sources, which makes reliably simulating the correct physics involved in a timely fashion a very strenuous task. Fortunately, there are several software tools available that already have addressed many of these issues, modeling a vast number of systems and its interactions within the operational environment. Therefore, instead of concentrating efforts on these models, we have opted for adopting a commercial off-the-shelf (COTS) solution to provide the simulation of the necessary systems within the desired scenario.

Using COTS simulation products allowed us to focus our research on the decision model employed by the enemy to manage its system-of-systems (SoS), which is comprised of the individual systems pertaining to the opposing air defense. We employed behavioral models of these systems as a means of reaching reliable depictions of the enemy operations, which support our threat evaluation and subsequent decision cycle. These processes are represented using probabilistic graphical models that leverage prior knowledge regarding the striker aircraft performance, as well as its current state, to assess the sequence of actions (i.e., the air mission plan) that would enhance the likelihood of meeting the mission objectives while reducing the threat level it is subjected to. At the present state of our research we are developing these models using Bayesian networks, and plan to expand to Multi-Entity Bayesian Networks (MEBNs) in the near future. We also intend to explicitly capture the decision process in the system by using Influence Diagrams and Multi-Entity Decision Graphs (MEDGs).

An important assumption of our current models is that the attack mission is being conducted by a single aircraft, which is rarely the case. This was made to simplify the threat assessment process, since it does not have to include threat prioritization methods and other aspects present in multiple aircraft sorties that would not add value to our current simulation goals. For instance, coordination between aircraft would require a more complex route planning process – since each aircraft poses constraints to other aircraft – which would also require a deeper knowledge on particular airstrike doctrines. In other words, the added complexity of modeling multiple aircraft would not sensibly affect the results we aim to collect, which explains the fact that this approach has been adopted in similar works (e.g., [4]).

The goal of this research is not to develop an automatic route generator such as proposed in [5, 6], but to provide a means to evaluate whether a particular route is viable for accomplishing the mission, and provides sufficient survivability rates. Another important point regarding our work is that the analyses take place during planning time and are not initially intended to cover events happening during mission execution (e.g., opportunity targets and unforeseen events). Nevertheless, analysts could adapt the models to include newly obtained information and suggest real-time in-flight route modifications to the pilot.

This chapter is organized as follows. Section 20.2 describes the goal scenario that guided the development of the proposed methodology. Section 20.3 brings a detailed description of the methodology, including the route evaluation process, threat assessment, and uncertainty representation. Section 20.4 contains the results of the simulation runs of a simplified scenario that serves as a proof of concept. Section 20.5 presents a discussion on some of the preliminary results already obtained and on the future developments of the approach. Finally, Sect. 20.6 presents a summary of the paper.

20.2 Goal Scenario

To guide our development, we adopted a scenario that consists of an unstable diplomatic situation between two fictitious countries: Westland and Eastland. To make the scenario more tangible, we assumed these two countries to be located in northeastern Brazil, and having a similar infrastructure with the one currently in place there, including its urban centers. The scenario also includes a background story of rapidly escalating tensions between the two countries, mostly because of an increasing belief by the Eastland Intelligence Agency (EIA) that Westland is using

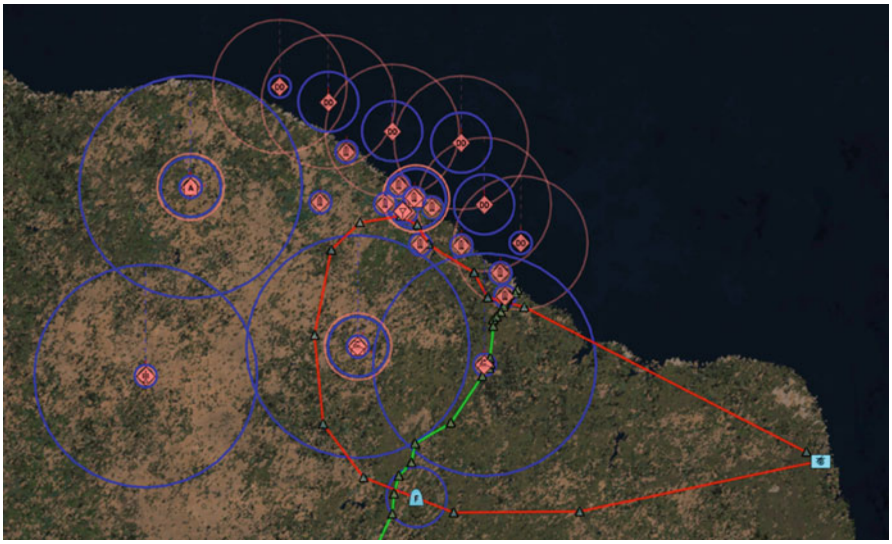


Fig. 20.1 Westland × Eastland scenario in VR-Forces

its uranium enrichment plant to produce weapons-grade isotopes. This works as a cover story to justify a strike in Fortaleza (capital of Westland in the scenario), as well as all the movements, weaponry, and assets involved in the simulation.

The main simulation in this part of the research emulates the situation in which Eastland designated a single Dassault Mirage 2000 aircraft, equipped with a joint direct attack munition (JDAM), to execute the airstrike departing from the city of Natal (capital of Westland in the scenario). The main idea is to avoid the opposing radar systems and take advantage of the surprise effect, since even with the increasing tension Westland would not be expecting such an early attack. Plus, even upon detection, Westland may hesitate in shooting the aircraft down, as it would mean an immediate declaration of war.

Figure 20.1 shows the scenario on VR-Forces, the COTS software utilized throughout our research so far. VR-Forces is a simulation environment developed by VT MÄK for scenario generation [7]. In addition to its graphical interface (front-end), VR-Forces consists of a back-end application, which is its actual simulation engine. Both VR-Forces front-end and back-end can be either embedded into another application or extended through plug-ins using the C++ API provided.

Abdellaoui et al. [8] provide a study comparing computer-generated forces (CGF) simulation software in terms of autonomy, learning and adaptation, organization, realism, and architecture. VR-Forces was considered to be the most suitable as a development platform, mostly because of its built-in Artificial Intelligence (AI) capabilities, very good documentation, responsive technical support, and its support for data logger export. The same conclusion was drawn by [9], which conducted a much more detailed analysis.

20.3 Methodology

The air mission methodology relies on three basic features: enemy threat assessment and response, air defense uncertainty representation, and route evaluation. The first two are directly related to the model of the scenario, since they can guide the behavior of the simulation agents as well as determine their positions and capabilities. The third, however, focuses on providing the metrics for the planners regarding the mission vulnerabilities and chances of success.

20.3.1 Enemy Threat Assessment and Response

Our prototype is not meant to handle automation of the planning process. Instead, its objective is to evaluate the planning done by humans (pilots and mission planners). In this context, the goal of the threat assessment part of the proposed system is to enhance the fidelity of the simulation behavior of the opposing forces. This is a departure from the approach adopted by most of the current adversarial simulations applied to mission planning systems, in which the hostile units would only perform rule-based actions, thus failing to encompass some of the inherent uncertainties of the scenario. This is especially true in cases that may not only involve military decisions, but also political ones.

The main goals of our threat assessment model are to define whether a given enemy unit poses a real threat, what would be the most effective response to that unit, and what available assets should be used in such response. Our prototype system is an initial attempt to reach those goals within a limited scope, and at this stage its main focus is on properly connecting a physics-grounded, high-resolution simulation with a behavioral model that mimics the environmental reactions while addressing uncertainty in a sound fashion. To achieve these goals and scope, we developed a simple Bayesian network (BN) model that captures the stochastic responses within the environment described in our scenario.

The first development challenge we had was to insert the BN model into a complex simulation environment such as VR-Forces, which mostly relies on rule-based approaches for AI. Further, we knew that after this initial model we would be seeking more complex responses that would likely demand increasing the sophistication of the probabilistic models. Among these possibilities is the use of Influence Diagrams to replicate decisions [10], or of Multi-Entity Bayesian Networks [11] to enhance the expressiveness of the probabilistic model. As a means of ensuring our solution could be extended with these techniques, we decided to develop our models using UnBBayes, an open-source, Java-based, probabilistic graphical framework developed by the Artificial Intelligence Group (GIA) from the Computer Science Department at the Universidade de Brasília [12]. UnBBayes has a GUI and an API that provides support to various algorithms and techniques via a plug-in-based architecture. This includes, but is not limited to, Bayesian inference,

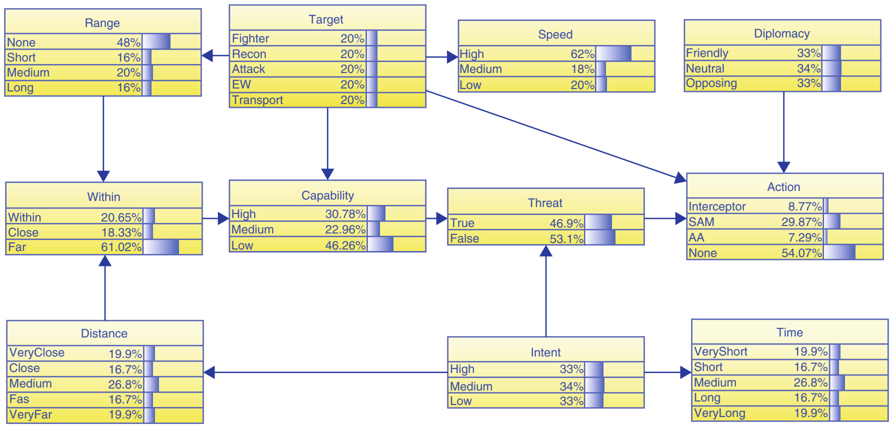


Fig. 20.2 Bayesian network for threat evaluation and response in UnBBayes

sampling, learning and evaluation, which brings some advantages compared to the other available software [13]. Figure 20.2 shows part of our Bayesian network model developed using UnBBayes.

The BN depicted in Fig. 20.2 follows the modeling approach proposed in [14], in which the variables present in an air defense scenario would be partitioned into proximity parameters, capability parameters, and intent parameters. Proximity parameters are those related to the distance between the aircraft executing the mission and its target (i.e., the defended asset, from the perspective of the model). In the BN depicted in Fig. 20.2, the proximity parameter is the *Distance* node. Capability parameters are related to the ability of the enemy's air defense system to inflict damage to the defended asset. In our model, these parameters are represented by the BN nodes *Range*, *Target*, *Time*, and *Speed*. Finally, Intent parameters refer to the enemy's intentions toward the defended asset. In our model, the node *Intent* has this goal.

In addition to the parameter nodes, our model also includes the nodes *Within*, *Capability,* and *Threat*, which represent the results of the interactions between the parameter nodes. These interactions should represent some of the observed characteristics of the system. For instance, the threat posed by a very distant target should be close to minimum, and increase with the reduction of this distance [15].

Based on the results obtained so far, this work extends the methodology in [14] by proposing another node type, the Pre-condition parameters. The goal with this parameter group is to address an important aspect for high-resolution, physics-grounded models, which is the need for considering external conditions that transcend the system's inherent characteristics. In Fig. 20.2, Pre-Condition parameters are represented by the node *Diplomacy*. The interaction of this pre-condition with the threat evaluation results in a node called Action that recommends the use of a system within the air defense SoS to address the identified threat.

Pre-condition parameters have to be defined prior to the simulation, since they carry key information about the system behavior that should dictate how the simulation would run. In our model, the Pre-Condition node *Diplomacy* conveys a probabilistic assessment of the predisposition of Westland to perform a prompt military response. The remaining nodes in the model represent evidence gathered from the scenario, providing information on the threats and the available options for dealing with it.

As we extend our model, we plan to adopt a similar approach to perform the "range check" while considering the opposite side's potential reaction according to the range. That is, the typical reaction of the enemy's air defense system as range changes will be computed, allowing for our own system to generate a recommended action (including the "no action" option). Implementing this capability, however, is highly dependable on the ability of gathering information from VR-Forces, as discussed in Sect. 20.4.

20.3.2 Air Defense Uncertainty Representation

On the mission planning process, since it is assumed that the pilots are provided with the best intelligence information available, a rather straightforward approach to deal with the inherent uncertainties regarding the locations and capabilities of the enemy's air defense systems is to simply ignore them [4]. This has been done in the first experiments of this research. However, in the current phase of the development, we are experimenting with two other approaches for dealing with uncertainty at the system's input.

The first approach focused on the location of the opposing assets based on terrain features, which are provided by preconfigured maps on VR-Forces. The most likely positions for the hostile units are assessed by considering the availability of reliable intelligence, even if such evidence is not new. Examples include satellite images retrieved sometime prior to the development of the mission plan. In this case, the location of the units identified on the image is limited by their speed and their design characteristics to overcome the surrounding terrain, which allows for generating probability heat maps for their location.

The second approach we have been experimenting with is suitable for the cases in which no explicit information concerning the enemy forces' capabilities and locations – such as the aforementioned satellite imagery – has been provided. In this case, intelligence analysis becomes the basis for assessing which areas possess the greatest chances of being defended due to their strategic value. The valuation methodology in this case may consider also intelligence information regarding the available units to be deployed by the enemy, the sites' accessibility, among many other factors. In short, a combination of SME assessment of enemy's intent, behavior, and capabilities forms the basis for developing a probability map, which will contain a proper representation of the intelligence process uncertainties.

It is important to remind the basic assumption that these uncertainties are accounted by the pilots themselves when they determine the mission routes to be followed and not by the proposed evaluation system, so the approaches listed earlier should be seen as a methodology for replicating the same level of situational awareness available to current systems. That is, these approaches provide a "good enough" replication of the tactical scenario for the purpose of properly generating the simulated hostile units and providing the required variability for the simulation runs.

For the experiments presented in this paper, the variability was obtained by employing random sampling from a predefined area that takes into consideration the units' ranges. In other words, it is assumed that the chance that the initial intelligence information is correct is equal to the chance of being wrong by an offset bounded by the unit's range on a given time window. This window represents the time elapsed between the information gathering and the planning process. Moreover, there is no consideration regarding the uncertainty of the systems' capabilities, mostly because this uncertainty is coupled with the location uncertainty, which is already included on the offset. In this context, future developments will aim to improve the estimation of these offsets and to limit the weapons and sensors ranges. The latter could be previously known, thus eliminating one of the coupled uncertainties.

20.3.3 Route Evaluation

The methodology proposed in this work allows for a lower level of complexity with respect to the route evaluation process. It differs from traditional mathematical models in that the underlying calculations are performed within the simulation software. Therefore, rather than having to develop complex models for each and every opposing system, as well as for the interactions between these systems, the process relies on premodeled parameterized units. The simulation of these units resembles an agent-based approach to modeling, since each unit individually possesses a set of characteristics and simple rule-based behaviors that collectively lead to the execution of a realistic, complex scenario.

The experiments suggest a reasonable level of convergence to stable parameters. Thus, with sufficient simulation runs it is possible to define probabilities for both mission accomplishment and survivability. Similarly to [4], the simulation results are traceable, meaning that the exact location is determined by the software, clearly indicating a vulnerable route section or waypoint.

To deal with the immediate limitation of assessing whether other waypoints along the route also pose a high vulnerability – in the case the aircraft is shot down on previous waypoints – the analyst can run a new batch of simulations disregarding the opposing units that previously engaged the aircraft. In this case, the best practice is for the pilot to change the previous waypoint to a less vulnerable one, unless in cases where the rules of engagement and the criticality of the mission impose a given route, even if it is highly vulnerable. In real operations, the latter can be attained by heavy reliance on threat denial systems such as electronic-warfare aircraft, SIGINT missions, etc. Currently, these are not explicitly modeled in this work, but their effect can be mimicked by accepting a higher-risk route in spite of the system's recommendations.

20.4 Simulation Results

Since the scenario presents a highly complex set of entities with multiple interactions amongst them, the authors developed a proof-of-concept prototype in order to generate initial results that would allow for preliminary analysis as well as evaluation of the simulation scripts. This section explores some of these initial results obtained by running the simulation with different types of approaching aircraft.

The proof of concept involves a scenario where an aircraft tries to fly over a defended region. The defenses are antiaircraft artilleries (ZSU-23 Shiika), SAMs (SA-6 Gainful), and interceptor aircraft (SU-27 Flanker). These assets can be activated by the air defense system, in which typical reactions are modeled through the probabilities generated by the Bayesian network.

The invader aircraft used were: A-10 Thunderbolt (Attack), EA-6B Prowler (Electronic Warfare), E-3 Sentry (Reconnaissance), F/A-18 Hornet (Fighter), and C-130 Hercules (Transport). All threats were considered to be active and were detected on a medium distance. This information is automatically set as evidence at the Bayesian network through external databases obtained from the simulation environment.

20.4.1 Fighter

The usual responses for fighters were either SAM or interceptor. Based on the Bayesian probabilities obtained, both responses were not very effective in neutralizing the threat. However, those results were a consequence of the air combat performed by the interceptor and the fighter. As expected, when successful flight tactics were employed, the interceptor could destroy the invader in a few instances. Also, the evasion tactics utilized by the fighter were sometimes ineffective to circumvent the SAMs (Fig. 20.3).

20.4.2 Attack

Attack planes usually fly in lower speeds than interceptors, resulting in a higher number of successful engagements by the latter. In addition to fighters, SAMs and

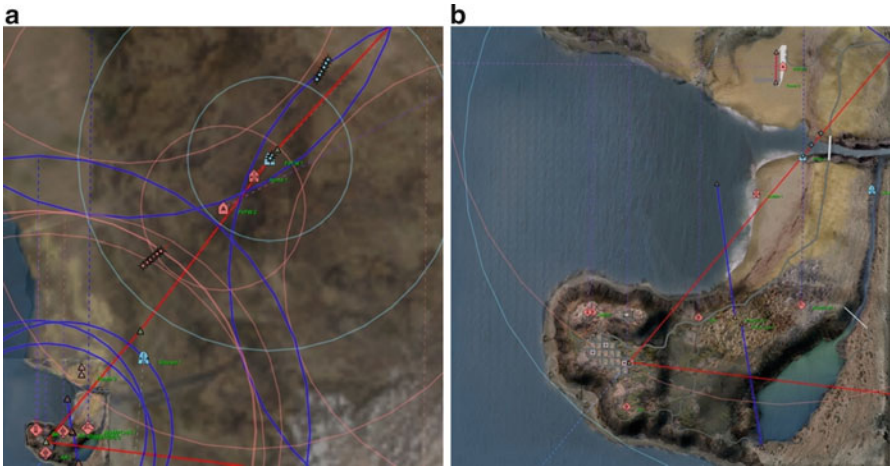


Fig. 20.3 (**a**) Fighter threat with interceptor response (**b**) Fighter threat with SAM response

interceptors were the most common responses, and yielded better results. Almost the totality of the runs recorded a successful SAM engagement as represented in Fig. 20.4.

20.4.3 Electronic Warfare and Reconnaissance

The Bayesian network results also indicated that the most used approach to engage either an electronic warfare or a reconnaissance aircraft was the employment of SAMs. On the great majority of runs these engagements were successful Fig. 20.5.

20.4.4 Cargo

The cargo plane was able to go through the defenses in the majority of the simulation runs, mostly because the antiaircraft artillery was not able to take the aircraft down. The main reason for this result was that the employed model for the artillery fire did not take into account the plane's speed, causing the projectiles to be always behind the aircraft. Even on very low altitudes and speeds, the AA was not capable of hitting the transport plane, as showed in Fig. 20.6.

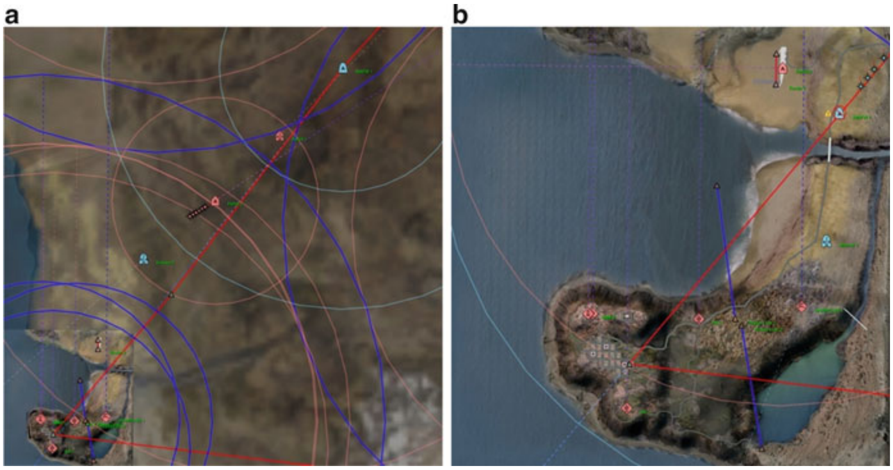


Fig. 20.4 (**a**) Attack aircraft threat with interceptor response (**b**) Attack aircraft threat with SAM response

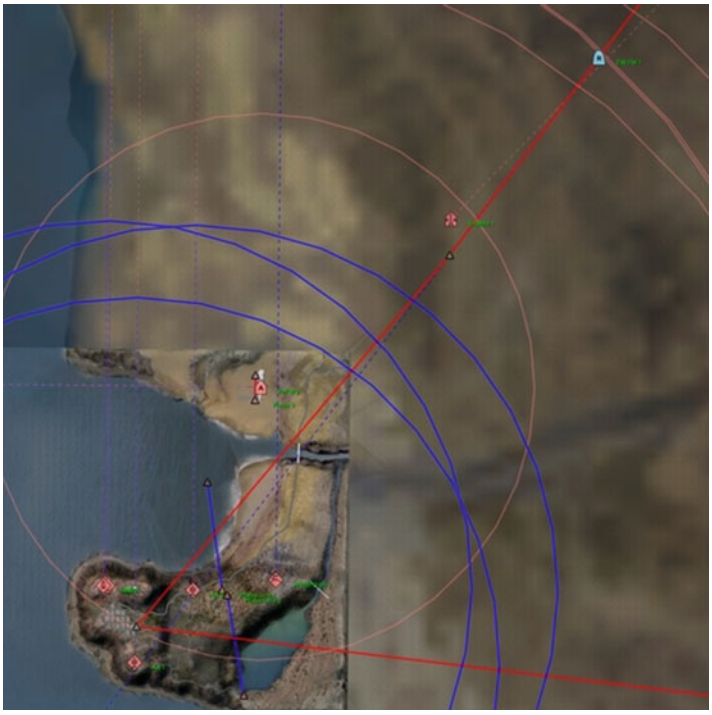


Fig. 20.5 Reconnaissance aircraft threat with SAM response

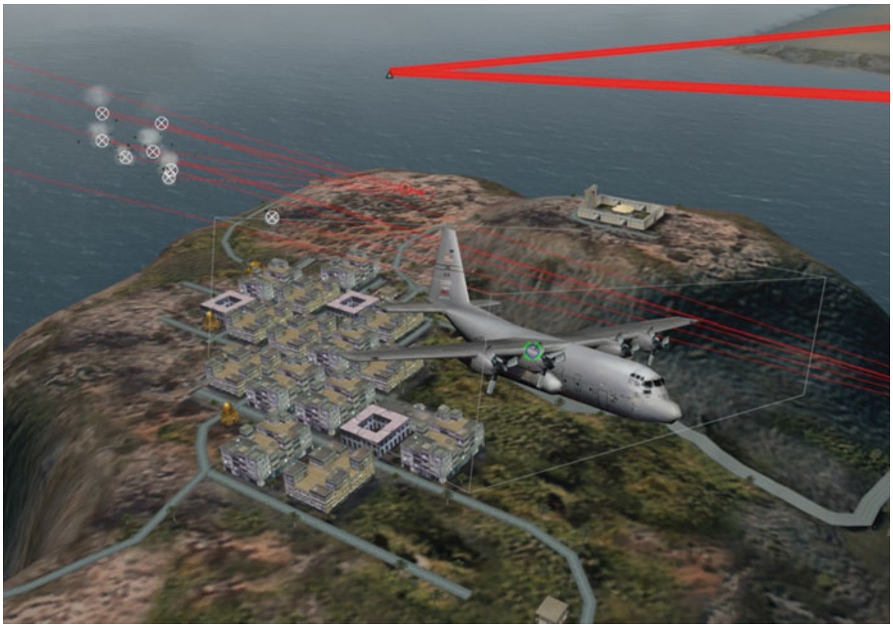


Fig. 20.6 Transport aircraft threat with AA response

20.5 Discussion and Future Work

Even though the interface between UnBBayes and VR-Forces is already implemented as part of our research and is working properly, some issues regarding the sensor output generation still persisted. This forced the data collection to be done manually, and caused considerable delays in the schedule of the experiments. An example of such issues is that the initial model did not consider possible interactions of the sensors with the terrain, which generated a series of blind spots. Hence, all the simulation runs would result on a successful mission, with not a single weapon fired against the attacking aircraft. These interactions can be evaluated through an intervisibility fan that shows visibility lines from a sensor to all the directions within its range as displayed in Fig. 20.7.

Furthermore, the manual setting prevents tests from being executed in batches, thus increasing the data collection time considerably. Apart from these caveats, the level of computational performance observed in each simulation run was very promising. Although a rigorous performance evaluation experiment has not been done, it is clear that the UnBBayes response has consistently stayed below 1 s. In terms of inference under a complex scenario, the Bayesian network model has been keeping a consistent and coherent output to VR-Forces, allowing for short and steady simulation runs. Current work is being made in order to define and test new

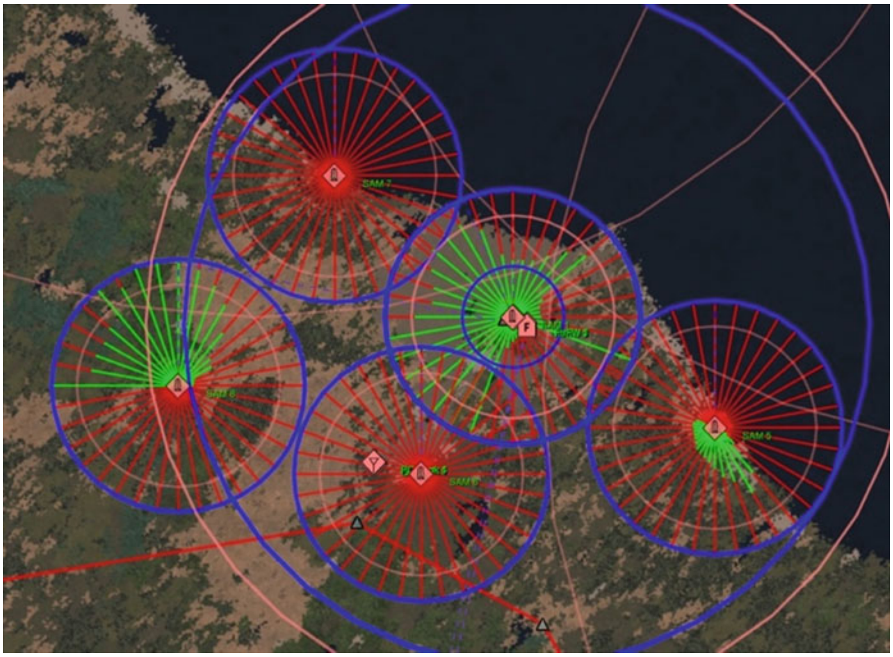


Fig. 20.7 Intervisibility fans in VR-Forces

variables, taking into consideration the computational time as well as the probability elicitation, which may increase complexity outside the performance thresholds with the increasing number of variables.

Finally, the authors are implementing the two approaches to uncertainty mentioned in 20.3.2., which are already adding the capability of batching, since it provides an automatic scenario generation process. Another improvement currently in development is the addition of road data, which are needed for calculating possible paths. This will allow for a better estimation of the movements of units on the map, which could also be simulated through VR-Forces, taking advantage of its ground movement behaviors.

20.6 Conclusion

This paper proposed a probabilistic-based simulation approach for expanding current threat assessment models within a decision support system, whose goal is to improve the mission success rate. The system assesses the likely responses to the air plan actions based on an enemy behavioral model, which have been historically implemented with rule-based systems. The methodology focuses on the attacker aircraft capabilities and intent, and introduces a threat response framework that includes pre-conditions and information regarding the air defense systems' capabilities. In addition, it lays out a scenario that contextualizes the method and provides the basis for evaluating the methodology.

To overcome unnecessary complexity, we opted for avoiding focusing on the automatic determination of routes, while shifting the research work to the decision aspects associated with the defender. One important bonus of this approach is that it leaves the route decisions to the planner. That is, with the information provided through this approach, the planners are well positioned to reassess the plan and make adaptations to reduce the vulnerability of the route. We plan to extend the work to incorporate these decision aspects within the uncertainty model, which would bring a new set of automation capabilities to the framework.

Acknowledgments The authors thank Danny Williams, from VT MÄK, who was instrumental in providing technical expertise in the VR-Forces suite and Shou Matsumoto for the support on its integration with UnBBayes.

References

1. Schulte A (2002) Cognitive automation for tactical mission management: concept and prototype evaluation in flight simulator trials. Cogn Technol Work 4(3):146–159
2. Theunissen E, Bolderheij F, Koeners GJM (2005) Integration of threat information into the route (re-) planning task. In: 24th Digital Avionics Systems Conference, Washington, DC, USA, vol 2, 14 pp
3. Richards MA, Scheer J, Holm WA (2010) Principles of modern radar. [electronic resource]. Scitech Pub., c2010, Raleigh
4. Erlandsson T (2014) A combat survivability model for evaluating air mission routes in future decision support systems
5. Quttineh N-H, Larsson T, Lundberg K, Holmberg K (2013) Military aircraft mission planning: a generalized vehicle routing model with synchronization and precedence. EURO J Transp Logist 2(1–2):109–127
6. Erlandsson T (2014) Route planning for air missions in hostile environments. J Def Model Simul Appl Methodol Technol. doi:10.1177/1548512914544529
7. VR-forces: computer generated forces – VT MÄK. [Online]. Available: http://www.mak.com/products/simulate/vr-forces. Accessed: 14 Aug 2016
8. Abdellaoui N, Taylor A, Parkinson G Comparative analysis of computer generated forces' artificial intelligence
9. Parkinson G (2009) AI in CGFs comparative analysis. Defence R&D, Ottawa, Canada, Summary Report, December
10. Poropudas J, Virtanen K (2009) Influence diagrams in analysis of discrete event simulation data. In: Winter Simulation Conference, Austin, Texas, pp 696–708
11. Laskey KB (2008) MEBN: a language for first-order Bayesian knowledge bases. Artif Intell 172(2):140–178
12. Matsumoto S et al (2011) UnBBayes: a java framework for probabilistic models in AI. In: Ke Cai (ed) Java in Academia and Research - ISBN: 978-0980733082. Annerley, Australia, iConcept Press Ltd, p 34

13. Mahjoub MA, Kalti K (2011) Software comparison dealing with Bayesian networks. In: Liu D, Zhang H, Polycarpou M, Alippi C, He H (eds) Advances in neural networks – ISNN 2011. Berlin Heidelberg, Springer, pp 168–177
14. Johansson F, Falkman G (2008) A Bayesian network approach to threat evaluation with application to an air defense scenario. In: 2008 11th International Conference on Information Fusion, Cologne, Germany, pp 1–7
15. Okello N, Thorns G (2003) Threat assessment using bayesian networks. In: Proceedings of the Sixth International Conference of Information Fusion, Cairns, Australia, vol 2, pp 1102–1109

Chapter 21
Tradespace Exploration: Promise and Limits

Paul D. Collopy

Abstract Tradespace exploration has become a trusted tool for conceptual design of complex engineered systems. It serves several purposes in the early phases of design, and can be very effective, particularly in comparison with previous methods and tools. However, tradespace exploration also has its limitations, which are not recognized or appreciated in the literature. Results are bounded by the design generator that populates the tradespace. The sampling of the design space can be very thin even when a large number of designs are generated. And the Pareto frontier does not warrant the trust that is placed in it.

Keywords Tradespace exploration • Pareto frontier • Conceptual design

21.1 Introduction

When one is immersed in systems engineering every day, it can be difficult to identify sea changes that take place on decadal scales. However, the turn of conceptual design toward tradespace exploration is such a change. From deep roots in response surface analysis [1] in the 1950s and optimization via design steering [2] in the 1990s, tradespace analysis became a technique of its own in the early 2000s. Although many researchers participated in this development, and the Aerospace Systems Design Lab at Georgia Tech deserves particular mention [3], it was Timothy Simpson and Michael Yukich at Pennsylvania State University with a practical multidimensional tool [4] and Adam Ross at the Massachusetts Institute of Technology with his emphasis on utility theory [5] that brought tradespace exploration into wider consideration.

Now tradespace exploration is regarded as a critical tool for managing the development of complex systems. Future system developments intend to bank on tradespace exploration to produce effective, affordable, and robust designs [6].

P.D. Collopy (✉)
University of Alabama in Huntsville, Huntsville, AL, USA
e-mail: paul.collopy@uah.edu

A.M. Madni et al. (eds.), *Disciplinary Convergence in Systems Engineering Research*, https://doi.org/10.1007/978-3-319-62217-0_21

21.1.1 What Is Tradespace Exploration?

At the risk of oversimplifying, tradespace exploration is a process by which a large number of alternative designs of the same system are automatically generated and graphed against two or more objectives. Here I use "objective" in the sense of Keeney-Raiffa [7], in which an objective is a collector of a set of related attributes of a system. Attributes are measurable properties of the system design that are meaningful to the user, owner, or maintainer (Keeney and Raiffa define attribute more broadly because they are interested in a very broad range of decisions, but we are only concerned with system design decisions). In the sense of Simon [8], attributes describe the face that the system presents to its outer environment, the world in which it is used.

Tradespace exploration uses objectives rather than attributes for two reasons. First, even a cursory description of a system design typically requires 10–20 attributes, and visualizing a system in these dimensions presents daunting challenges [9]. Second, trades between attributes within an objective are often solvable by widely accepted analytic methods, whereas trades between objectives present the sort of difficulty that tradespace exploration is intended to mitigate.

Figure 21.1 presents an example tradespace plot in two dimensions, with the objectives essentially being cost and accuracy. The system is a missile interceptor, specifically a missile that is designed to shoot down a nuclear warhead traveling on a ballistic trajectory in near space beyond the atmosphere. The horizontal axis is cost, and the vertical axis is accuracy, measured as the probability that the interceptor will disable the target warhead. Note that the data have been falsified because actual data on such a system are protected, and the example is only used for illustration.

Each of the triangles in Fig. 21.1 represents a system design, and the graph of the triangle indicates the cost and accuracy of the design. There is no distinction among designs except the cost and accuracy. The triangles are created by a design generator that constructs many system designs according to a preset pattern of configurations, uses simulation to evaluate the attributes of each system, and aggregates the attributes into objectives.

Here the unit cost is an objective that collects attributes such as unit manufacturing cost, transportation and basing cost, development cost, and maintenance cost. The aggregation of these costs is prescribed by the methods of life cycle cost analysis [10], and is allocated on a per interceptor basis. The accuracy measure, Unit Probability of Intercept, aggregates all performance measures, such as command and control, tracking, guidance, maneuvering, and so on.

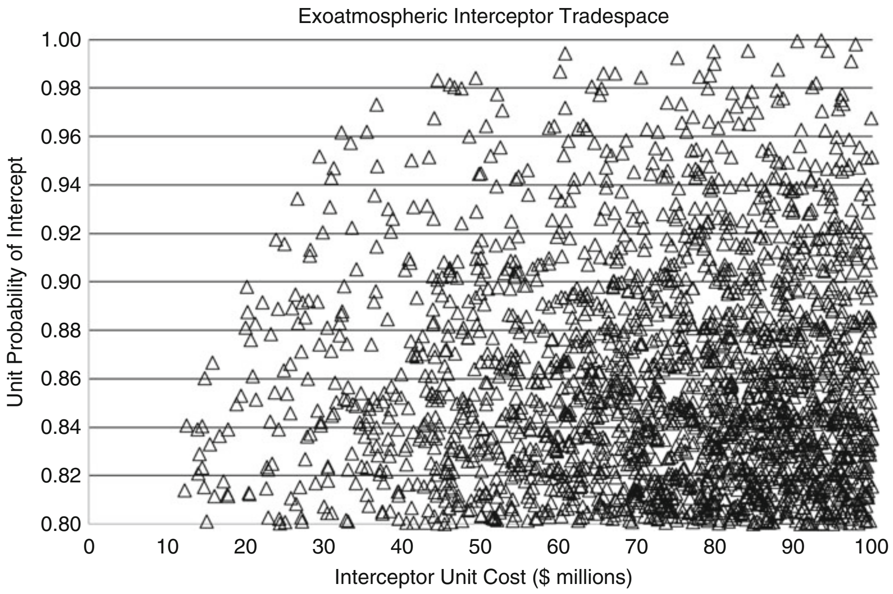


Fig. 21.1 Example tradespace (false data used as an example)

21.2 Benefits of Tradespace Exploration

21.2.1 Knowledge Development

There are several uses for tradespace exploration. The least controversial is knowledge development, the exploration of the impact of design choices, and the appreciation of the bounds of feasible design [11]. Designers and users can play with the tradespace exploration tool, making changes and observing the effect, or focusing in a particular design region and generating a dense set of designs. This is the same activity that an earlier generation of engineers pursued with carpet plots, portraying the relationship between design variables and attributes in as high a dimension as pencil and paper sketches would permit.

21.2.2 The Concurrent Development of Design and Elucidation of Preferences

It has been noted in the literature [11], and it is certainly true in the author's experience, that when an objective function is used to search for designs, the designs that are recommended are often repugnant to the very person who constructed or informed the objective function. Generally, this dilemma results in the discovery of an attribute that is important, but has not been included in the objective function. For example, a submarine might be designed for low cost and high top speed at a depth of 100 meters. The tradespace exploration yields a set of designs that all have very poor reliability. Thus, it becomes apparent that reliability was neglected in the formulation of the optimization problem and must be added to the objective function.

An extension is the design-by-shopping paradigm [12]. Here the designer explores the tradespace and learns what he (or she) likes at the same time that he learns what designs are possible. The exploration leads to a well-developed objective function design and elucidation of preferences and a conceptual design at the same time. Or a design is selected without bothering to express an objective function.

21.2.3 Conceptual Design

Although downplayed in the literature on tradespace exploration methods, many researchers use tradespace exploration to identify promising designs [13]. The usual procedure, as illustrated in Fig. 21.2, is that a Pareto frontier is constructed (the dashed line), which is a set of designs in the tradespace that contain the optimal design for any possible weighting that might be assigned to the tradespace objectives in the construction of an objective function. Essentially, we are given that, all else being equal, moving upward on Fig. 21.2 improves a design, and moving leftward also improves a design. So the best region on the plot is the upper left corner. But the direction of goodness is not clear. Although an example arrow is shown, the Pareto frontier assumption is that the direction of the arrow could be anywhere between pointing left and pointing upward. Whatever the direction, the design furthest in that direction must be one of the 13 designs that fall on the Pareto frontier, as they lie further left of the points of similar probability and further up from the points of similar cost. Further, it is observed that the point of greatest curvature on the Frontier is the circled triangle. The circled design is identified as sitting on the knee of the curve and is selected as the preferred design. The maximum curvature point, or knee, is privileged because, if the direction of the ideal arrow is uncertain, and if the direction can be described as a random variable uniformly distributed between 270° and 360°, then C is the point most likely to be optimal.

21.2.4 Advancing the State of the Art

There are three areas in particular where tradespace exploration has pushed forward the state of the art in conceptual design of engineered systems:

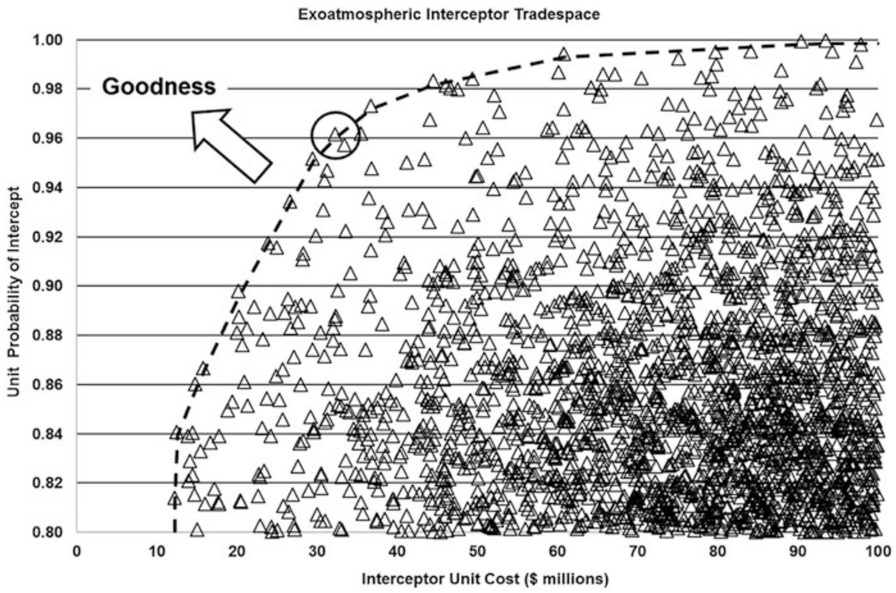


Fig. 21.2 Pareto frontier with the optimal design circled

- Tradespace exploration moves away from a naïve strategy of distilling requirements from stakeholders to a more sound strategy of examining preferences and studying possible designs prior to, or instead of, formulating requirements.
- Tradespace exploration offers the design team many possible designs rather than quickly focusing down to one or three designs.
- Tradespace explorations appreciates and exploits the interrelationship between customers or designers seeing designs and forming preferences with regard to design attributes [11].

21.3 Hazards of Tradespace Exploration

In summary, the danger of tradespace exploration is that, to the untutored user, it appears to do much more that it is actually doing. I will examine four particular aspects of this danger:

- The tradespace is bounded by the limitations of the design generator.
- The tradespace can only show a very thin sample of the possible designs.
- The knee on the curve can be placed pretty much anywhere on a Pareto frontier – it is an arbitrary selector of best designs.
- Utility on one axis is not really utility.

21.3.1 Bounded by the Design Generator

Design is an innately human and creative process. We revere new technology as the implementation of wholly novel elements within the design of a system such as an aircraft or missile. However, a design generator cannot create. It can only rearrange a predefined design and step across limited ranges on a set of design variables. Frontiers are pushed forward by novel technical innovations, but design generators can only interpolate among pre-existing system structures fit to preset technologies.

It is possible to create a new component technology and demonstrate its benefit through tradespace exploration of systems utilizing the technology – in fact, this is a very good application of tradespace exploration. However, a design generator cannot innovate as needed. That is, when an engineer is hemmed in by a limitation in the current state of the art, he or she creates new designs and new technologies to break through the limitation. A tradespace design generator is necessarily stopped by the same limitation – it cannot innovate, cannot create a novel design solution. These limitations then define the Pareto frontier, which is the area of emphasis in tradespace exploration. As a result, the frontier is false, not representative of how far real designs could push.

21.3.2 A Very Thin Sample of the Possible Designs

When 5000 designs are arrayed on a tradespace, it is tempting to think that every possible design is displayed. However, because tradespaces are necessarily populated using automatic design generators, the range of displayed designs is necessarily quite limited. Even very smart design generators are restricted to:

1. Variations in finite set of design variables, typically less than 20, preselected by the creator of the design generator (such as aircraft planform area, wing sweep, chord, camber)
2. Ranges in the design variables, and combinations of design variables, within which the generator behaves reliably
3. Attributes that can be tractably estimated (such as aircraft fuel consumption at level cruise)

Tens of millions of design variables are in play in the complete design of a complex engineered system such as an aircraft or missile. It is perhaps more fair to compare tradespace exploration to preliminary design, but even here, hundreds of design variables are determined by the design team. A design generator takes an extremely thin slice through that 100 dimensional space. A good full factorial exploration of 10 options apiece across each of the 100 design variables would create one googol, 10^{100}, which is much more than the estimated number of atoms in the known universe. If a design generator were to create 10 million designs, that would be 0% of the true design space to any reasonable precision.

21.3.3 *The Knee on the Curve Can Be Placed Pretty Much Anywhere on a Pareto Frontier: It Is an Arbitrary Selector of Best Designs*

As already mentioned, tradespace exploration tends to focus attention on the design at the knee of the curve of the Pareto frontier, because, if direction of increasing utility is a random variable uniformly distributed over a range of angles, this particular design is the most likely to be the best. However (neglecting for a moment the philosophical conundra introduced by treating preferences as probabilistic), such a distribution is absurd. It would become nonuniform just by changing the scale of an axis, or changing the aspect ratio of the graph (e.g., switching portrait to landscape). Correspondingly, changing scales or aspect ratios will change which design is at the knee of the curve. Indeed, with enough manipulation of the display, most of the designs located anywhere on the Pareto frontier can become the favored design at the knee of the curve. Figure 21.3 is the same Pareto frontier as shown in Fig. 21.2, with only the aspect ratio slightly changed. Observe that a different design is now located at the knee of the curve.

The same shift can be accomplished by changing the scale of the vertical or horizontal axis without changing the aspect ratio. For example, if the vertical axis in Fig. 21.3 is changed from 0.8–1.0 to 0.9–1.0, it will have the same effect on the knee of the curve as changing the aspect ratio by a factor of 2.

The concern about which design is the best design in a tradespace can be alleviated very simply by displaying isovalue lines (curves on which every point has equal value) on the graph. The direction of increasing value is everywhere normal to the isovalue lines, so that it is easy to pick out the best design – not the design with the highest probability of being best, but the actual best design, for deterministic assessments. If each design is represented by a probabilistic cloud, the expectation of value can be shown within the cloud, in which case the best design is once again simply visualized using isovalue lines. Figure 21.4a shows isovalue lines on the same axes that the tradespace is plotted against, and Fig. 21.4b combines the Pareto frontier (dashed) with the isovalue line closest to the frontier (solid). Interestingly, the designs near the knee on the curve are found to be the worst designs on the Pareto frontier – a single counterexample that should totally discredit the knee-on-the-curve-is-best hokum.

These isovalue lines were generated from a value model developed for the Missile Defense Agency to assess missile defense technologies [14]. The model is as follows:

$$\text{value} = \frac{\text{cost}}{\ln(1 - P)} \tag{21.1}$$

Cost is unit cost (the horizontal axis in the example plots) and P is the probability of intercept (the vertical axis). In this model, value is always negative, so a decrease in the magnitude of value is an increase in value. The gradient of value (direction of

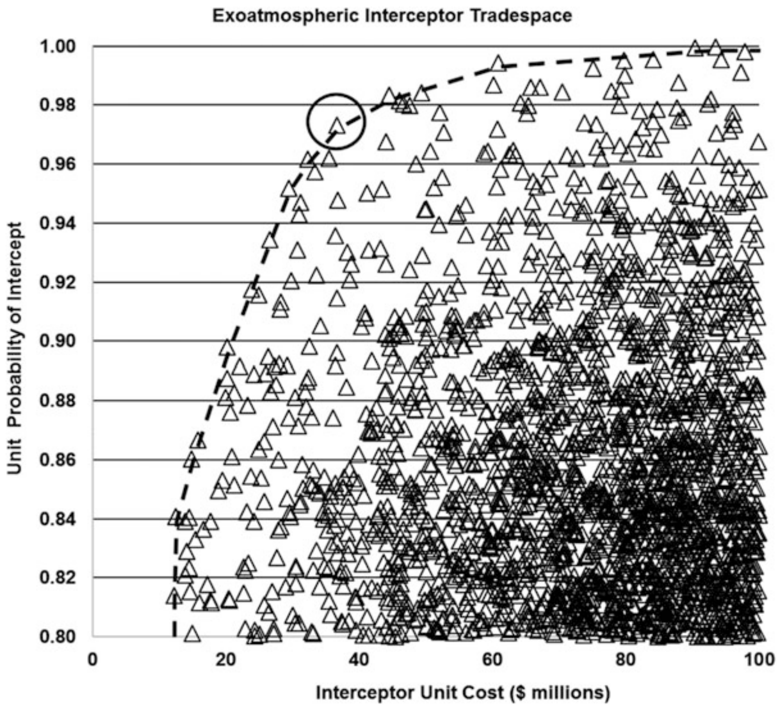


Fig. 21.3 Pareto frontier with the knee shifted

greatest increase) is always perpendicular to the isovalue lines. Visually identifying the best designs is straightforward – they are near the lower left corner of the graph.

In practice, isovalue lines are never plotted on tradespaces, although clearly they should be. The reason provided to this author for excluding value from tradespaces is that value hides the qualities of a design under a single scalar. Tradespace exploration practitioners prefer to roll up all the attributes of a design under two scalars, often utility and cost [15]. However, the isovalue lines in Fig. 21.4 hide nothing, while a utility axis can conceal quite a few attributes under a single aggregate.

21.3.4 Utility on One Axis Is Not Really Utility

The plotting of consumer utility versus cost is common in the simplistic Marshallian approach to microeconomics that is still taught in many undergraduate courses. However, the tradespace exploration literature provides references that clearly imply that utility of the sort introduced by von Neumann and Morgenstern [16], and proceduralized by Keeney and Raiffa [7], which has a meaning very different from what Alfred Marshall [17] and economists of the early twentieth

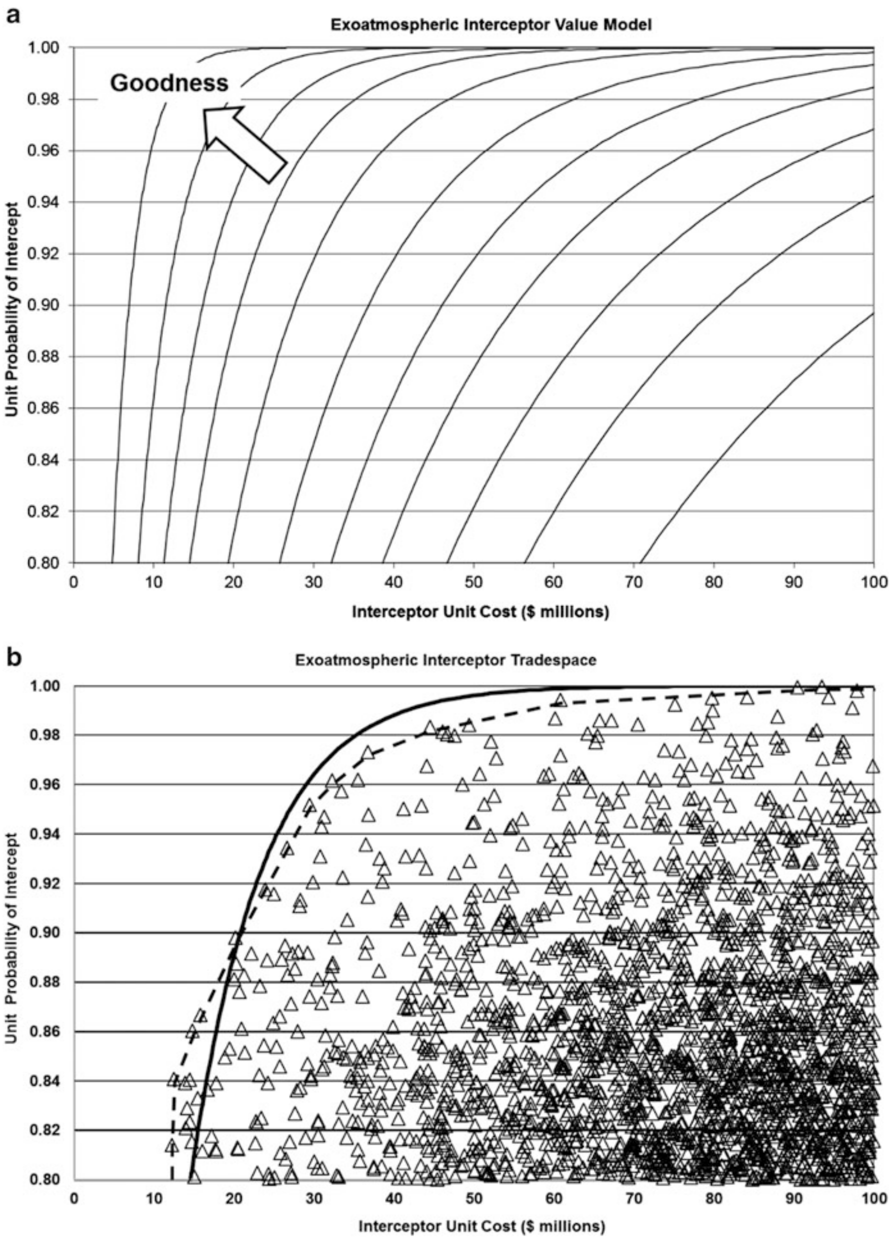


Fig. 21.4 Isovalue lines on a tradespace: (**a**) shows a set of isovalue lines from the missile defense value model; (**b**) places one isovalue line (*the solid line*) alongside the Pareto frontier (*dashed line*) in the tradespace. The best designs are *lower left*, on the good side of the isovalue line

century intended. This sort of utility is comprehensive, a measure of overall preference for the designs. A design that scores highest in utility ought to be the most preferred design. It is not utility of benefits or all utility except costs. If utility exists as an attribute orthogonal to cost (any form of cost), and if cost matters, then this is not von Neumann utility and the analysis does not conform to Keeney–Raiffa decision methods. In particular, one cannot apply risk aversion to benefits and not to costs, and then hope to make a rational composite decision, but this mistake is made when Marshallian utility is confused with von Neumann or Keeney–Raiffa utility.

Under the US White House Office of Management and Budget guidelines for US federal economic analysis [18], the axis orthogonal to cost would be labeled benefit. Then preference should be related to the amount by which benefit exceeds cost.

21.4 Conclusion and Discussion

Tradespace exploration is a valuable tool for exploring the effects of different designs on preferences and clarifying both what is possible and what is desired. It can inform system designers and stakeholders about the design characteristics of a system. However, it is not an adequate substitute for exploratory design, with actual engineers creating and innovating. Tradespace exploration cannot find the limits of capability and performance for a system. Tradespace exploration is constrained by its design generator to such an extent that, if you do not know who wrote the design generator used in the tradespace, you really have no idea what the tradespace means. Coverage of the tradespace and precision of the Pareto frontier are illusions. No tradespace tool can provide coverage or precision.

Moreover, the Pareto frontier itself cannot deliver the knowledge and insight that users expect. The knee on the curve of the frontier might be the best design in the tradespace, or it could be the worst design on the frontier. The knee may have been strategically placed at a particular design by the person who configured the tradespace plot. The location of the knee is not determined by the designs generated for the tradespace, but is instead an artifact of the axis scales and the aspect ratio of the plot.

Designs within a tradespace can be ranked rationally. One method is to plot isovalue lines onto the tradespace. However, a Pareto frontier alone will not provide a ranking of designs or any reliable direction as to which designs should be developed further.

Tradespace exploration can contribute substantially to improved system design. But as long as it is oversold and misused, it is likely to cause more harm than good in the design process.

Acknowledgments This material is based in part upon work supported by the National Science Foundation under grants CMMI-1161039 and CMMI-1447316.

References

1. Box GEP, Wilson KB (1951) On the experimental attainment of optimum conditions. J R Stat Soc B13(1):1–45
2. Winer EH, Bloebaum CL (2001) Visual design steering for optimization solution improvement. Struct Multidiscip Optim 22(3):219–229
3. Briceno SI, Mavris DN (2002) Quiet supersonic jet engine performance tradeoff analysis using a response surface methodology approach. SAE World Aviation Congress, Phoenix, November 5–7
4. Stump G, Lego S, Yukish M, Simpson TW, Donndelinger JA (2009) Visual steering commands for trade space exploration: user-guided sampling with example. J Comput Inf Sci Eng 9(4):044501
5. Ross AM, Hastings DE, Warmkessel JM, Diller NP (2004) Multi-attribute tradespace exploration as front end for effective space system design. J Spacecr Rocket 41(1):20–28
6. Spero E, Avera MP, Valdez PE, Goerger SR (2014) Tradespace exploration for the engineering of resilient systems. Procedia Comput Sci 28:591–600. doi:10.1016/j.procs.2014.03.072
7. Keeney RL, Raiffa H (1976) Decisions with multiple objectives: preferences and value tradeoffs. Wiley, New York
8. Simon HA (1969) The sciences of the artificial. MIT Press, Cambridge, MA
9. Jones CV (1996) Visualization and optimization. Kluwer Press, Boston
10. Eschenbach TG (2010) Engineering economy: applying theory to practice, 3rd edn. Oxford University Press, New York. ISBN-10: 0199772762
11. Miller SW, Simpson TW, Yukish MA, Bennett LA, Lego SA, Stump GM (2013) Preference construction, sequential decision making, and trade space exploration. Paper DETC2013/DAC-13098 in Proceedings of the ASME 2013 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference IDETC/CIE 2013, August 4–7, Portland
12. Balling R (1999) Design by shopping – a new paradigm? In: Proceedings of the Third World Congress of Structural and Multidisciplinary Optimization (WCSMO-3), Buffalo, pp 295–297
13. Lafleur JM, Saleh JH (2009) Exploring the F6 fractionated spacecraft trade space with GT-FAST, AIAA Paper 2009-6802. American Institute of Aeronautics and Astronautics, Reston
14. Collopy PD (2008) Value of the probability of success, AIAA Paper 2008-7876. American Institute of Aeronautics and Astronautics, Reston
15. Viscito L, Ross AM (2009) Quantifying flexibility in tradespace exploration: value weighted filtered outdegree, AIAA Paper 2009-6561. American Institute of Aeronautics and Astronautics, Reston.
16. von Neumann J, Morgenstern O (1947) Theory of games and economic behavior. Princeton University Press, Princeton
17. Marshall A (1980) Principles of economics, 1st edn. Macmillan, London
18. Anon. (1992) Guidelines and discount rates for benefit-cost analysis of federal programs, Circular No. A-94. Revised of the White House Office of Management and Budget, October 29. http://www.whitehouse.gov/omb/circulars_a094. Accessed 1 Nov 2016

Part IV
Model-Based Systems Engineering and Integration

Chapter 22
Model-Based Systems Engineering: Motivation, Current Status, and Needed Advances

Azad M. Madni and Michael Sievers

Abstract As systems continue to grow in scale and complexity, the Systems Engineering (SE) community has turned to the Model-Based Systems Engineering (MBSE) paradigm for managing complexity, maintaining consistency, and assuring traceability during system development. We distinguish MBSE, which is a holistic process comprising system specification, design, validation, and configuration management, from engineering with models, which has been a common practice for centuries. Even though MBSE is beginning to see a fair amount of use in a number of industries, several advances are needed on a number of fronts to realize its full benefits. The paper discusses the motivation for MBSE, models and meta-models, modeling languages, the role of modeling and simulation in SE, and the current state of maturity of MBSE. The paper concludes with a discussion of needed advances to realize the potential benefits of MBSE.

Keywords MBSE • Ontology • Meta-models

22.1 Introduction

Engineers have used models in a variety of forms for centuries, and "engineering with models" has been part of the systems engineering (SE) profession for decades. However, the increasing scale and complexity of systems have caused systems engineers to rethink system development. What has emerged is model-based systems engineering (MBSE), a new paradigm that places models at the center of the system development process. The term MBSE was first introduced by Wymore [30]. MBSE is different from "engineering with models" in important ways

A.M. Madni (✉)
University of Southern California, Los Angeles, CA, USA
e-mail: Azad.Madni@usc.edu

M. Sievers
Jet Propulsion Lab, California Institute of Technology, Pasadena, CA, USA
e-mail: msievers@jpl.nasa.gov

[24]. MBSE promises to be a more rigorous and effective means for developing complex systems [28]. In MBSE, the "model" is the sole source of truth and reflects both the state and status of system development. The model embodies multiple, complementary, and mutually compatible perspectives. One or more model perspectives come into play to answer questions posed by various stakeholders. This is in sharp contrast to engineering with models, where multiple models are employed often with inconsistent assumptions and underlying semantics. These issues, which have previously surfaced in the computer-aided systems engineering and concurrent engineering eras [9, 16, 18], continue to be an ongoing concern. The value of MBSE stems from the fact that all system-related information is stored in a central repository. This characteristic enables the interconnection of model elements and the ability to effectively retrieve desired information and reason about the system.

22.2 Systems Engineering, System Models, and Key Concepts

Volumes have been written on systems engineering and system engineering processes. At its core though, systems engineering distills into a few canonical functions: identifying stakeholders – individuals or groups who have a say in system development; identifying stakeholder needs and concerns (define issues or questions of interest); manage stakeholder expectations; identify system goals requirements, and boundaries; define external influences and relationships with external entities; set, maintain, and manage schedules and budgets; establish configuration management practices and maintenance processes; develop candidate design concepts and perform trade studies and sensitivity analysis to address stakeholder needs; conduct reviews; select and iterate a baseline to ensure coverage of stakeholder needs, and system requirements; create design descriptions, user documentation, and risk matrices for risk prioritization, mitigation, and management; monitor implementation, integration, test, and system acceptance.

Traditionally, the artifacts produced by these functions are captured in multiple text documents. Then, as designs change, these documents are updated and typically maintained in a project library. When there are several stakeholders, or when the system (or system of systems) is complicated (i.e., contains a large number of components) or complex (i.e., exhibits unanticipated and emergent behaviors), documentation invariably tends to be incomplete and inconsistent. Moreover, "language" differences among stakeholders can potentially lead to misunderstanding and incorrect implementation. For example, both controls engineers and computer engineers use "bandwidth" as a performance measure. However, "control bandwidth" specification is roughly a factor of 20 less than "operations per unit time" needed for implementing the control algorithm. Thus, without clear agreement on basic terminology and relationships, a system becomes susceptible to failure.

Development processes such as "waterfall," "spiral," "V," "hybrid," and "agile" have all been used in different contexts to implement canonical systems engineering processes and activities. These processes define the steps and artifacts produced at each step. MBSE, which supports these processes, can produce the same artifacts. However, as noted earlier, a model is a proxy for the real system when it comes to conducting design trades, analyses, validation, and user training. As the design progresses, abstract models become increasingly more concrete. Eventually, certain model components are realized as physical subsystems.

A system model is an abstract representation of reality that often integrates diverse inputs from different disciplines for a defined purpose. A model in model-based methods is a "living representation" of a system in that it evolves as details are progressively added throughout the system's life cycle. A collection of complementary and integrated model perspectives represents the sole "source of truth" about the system under development. These integrated perspectives are persistently stored in a repository and progressively refined to answer a greater number of increasingly more detailed stakeholder questions.

Based on the purpose of the model, a model may abstract or ignore certain features deemed unimportant or not relevant to the purpose of the model, or the needs of the stakeholders. This flexibility is essential because it allows stakeholders to view the system from different perspectives and in terms of specific features relevant to their needs. For example, a structural engineer might need to know the size, mass, and location of electronics boxes but not the detailed content of the boxes. Conversely, a software engineer does not need to know the size, mass, and locations of boxes, but does need to know their contents. A model-centric design can capture all these aspects of the system, while also providing stakeholder-specific views through lenses that focus on specific aspects of the model.

The purpose of models is to: facilitate understanding and offer insights; enable communication; support visualization; document decisions and rationale; enable verification and validation of system requirements, structure, and behavior; trace behavior to requirements; support marketing; and enable performance analysis (Fig. 22.1).

MBSE supports systems engineering functions, facilitates communication among stakeholders, and enables relevant analyses. These include: defining model scope; specifying system stakeholders and their concerns; establishing common terminology and relationships in the system domain; defining system goals, requirements, and behaviors; defining system structure, interconnections, and underlying equations; extracting stakeholder-specific views of the model; defining testing requirements and methods; linking entities within the model; and maintaining a model repository with the means to extract information for visualization, simulation, and analyses.

Employed appropriately, models can be a source of much needed insights. However, their inappropriate use can produce misleading results. The renowned statistician, George Box [4], famously declared, "Remember that all models are wrong; the practical question is how wrong do they have to be to not be useful."

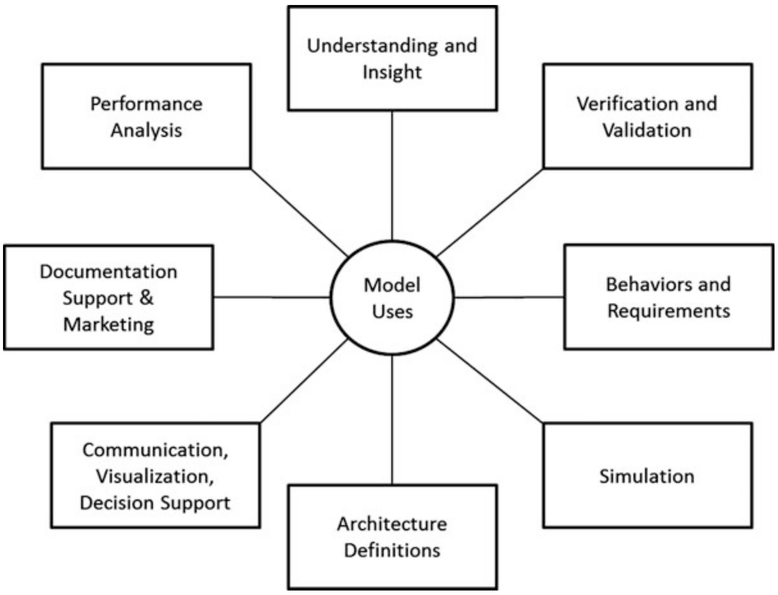

Understanding and Insight
Performance Analysis
Verification and Validation
Documentation Support & Marketing
Model Uses
Behaviors and Requirements
Communication, Visualization, Decision Support
Simulation
Architecture Definitions

Fig. 22.1 Representative stakeholder needs supported by models

Since then, most researchers and practitioners have pursued modeling with this awareness.

The two key concepts that are frequently associated with modeling are model verification and model validation. *Model verification* is performed to ensure that a model has been implemented correctly. *Correctness* implies completeness, consistency, and traceability. *Completeness* means that all elements, relationships, properties, inputs, processes, outputs, and constraints are sufficiently specified and reflect the needs of the stakeholders. The completeness of a model can be assessed based on whether or not a model is capable of answering the set of questions (so-called competency questions) that need to be answered at specific points in the system's life cycle. *Consistency* means the degree to which the model contains no conflicting requirements, assertions, constraints, functions, or components. *Traceability* implies that all concepts, relationships, and results originate in a specified requirement, standard, or guideline. A correctly implemented model is free of syntactic and semantic errors. In practice, there are limitations to model verification in that no computational model can ever be fully verified, and there is no way to guarantee absolutely error-free implementation. However, a high degree of statistical certainty can be achieved as a model is iteratively tested and corrected when errors surface. In principle, a properly constructed testing program can increase the level of certainty for a "verified model" to an acceptable level. A verified model is one that passes all the verification tests. Thus, the desired outcomes, the measures of effectiveness (MOEs), and the measures of performance (MOPs), need to be unambiguously defined, and measurable.

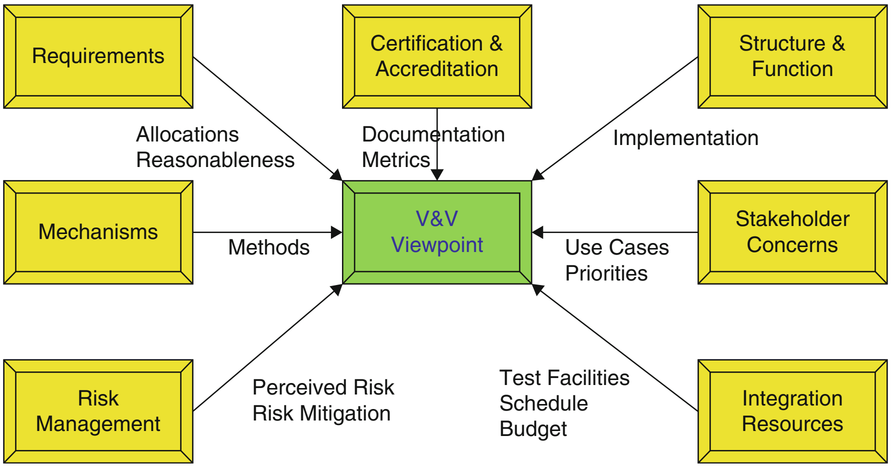


Fig. 22.2 Relationship of V&V to other development activities

Model validation confirms that the model complies with external requirements (i.e., theories, data, regulations, standards, and planned/desired behaviors). Model validation confirms the correspondence of the model with the real-world system (i.e., veridicality).

There is an implied assumption that validation tests are complete only with respect to the objectives of the model. In this regard, Active Nonlinear Tests (ANTs), a promising V&V technique [12], explicitly formulates a set of mathematical tests to "break the model." Each time a model fails a validation test, it is appropriately modified until it passes the test. This iterative process builds confidence in model validity. Being able to quantify the uncertainty in a model further enhances confidence in the model. Ultimately, there is no such thing as a completely validated model, or for that matter a completely validated system. Figure 22.2 depicts the relationship of V&V to other development activities.

A model can be represented using a modeling language, an algorithm, or parametric curves. The modeling language can be informal or formal. An informal model is a text description (prose), a concept diagram, or a cartoon (i.e., a stylized drawing with no formal semantics). A formal model relies on formalisms (e.g., state charts, sequence diagrams, Petri nets) and their visual representation. Formal models can be purely descriptive, or both descriptive and executable. Models of complex systems typically employ a combination of formalisms and visual representations that are typically augmented with text descriptions.

MBSE is motivated by the need to overcome specific deficiencies that adversely affect system architecture, design, and concepts of operations (CONOPS). For example, building the wrong features is a costly form of waste in engineering development [13]. In the same vein, some organizations tend to charge ahead with modeling without realizing that the original statement of the problem may not be the best, or even right [11]. It is invariably the case that the lack of a closely coordinated design can lead to downstream integration issues. An obvious fact that seems to be underexploited is that the greatest degree of freedom and the largest number of solution options exist when the problem is first defined. Thereafter, as design decisions are made, solution options narrow, establishing the bounds on life cycle costs. Some view MBSE as an effective, and possibly revolutionary, means to optimize speed, cost, and quality [28]. Early adoption of MBSE is beginning to show evidence of reduction in development time and error rates. In part, this can be attributed to developing a better understanding of the problem. According to Jorgenson [10], a traditional functional requirements decomposition approach is likely to capture 50% of problem understanding. However, employing operational concepts with use cases and scenarios can increase problem understanding to more than 90% the first time through. While hard numbers on development time reduction are hard to come by, early studies indicate that up to 40% fewer requirement defects were found with MBSE [14]. While current MBSE performance data are qualitative, quantitative metrics are being used to collect more conclusive evidence to support the MBSE value proposition [6].

Model-based methods in general, and MBSE in particular, are currently viewed within the SE community as keys to enhanced engineering effectiveness. MBSE is more than just the use of models in support of systems engineering. It is worth recalling that design engineers have been “engineering with models” for centuries [17]. However, these models seldom share assumptions and common terminology. Also, the interoperability/integration of these models tends to be ad hoc. In sharp contrast, MBSE takes a much more systematic and formal approach to systems modeling, integration, and communication. By exploiting a common system modeling language, MBSE circumvents potential disconnects among people and models arising from language differences. Also, MBSE requires assumptions to be made explicit, thereby reducing misunderstanding in communications.

The shift from document-centric to model-based approaches is currently underway. With model-based approaches, documents can be automatically generated from models using applicable model perspectives. Documents generated in this fashion reflect the prevailing state of the model. With model-based approaches, new information can be readily incorporated, and automated comprehensive traceability can be achieved.

Recent studies suggest that engineers spend inordinate amounts of time searching for and assembling information. Also, as systems continue to increase in scale and complexity, system requirements tend to increase dramatically. Therefore, trying to carry these requirements in mind, and managing them as an addressable checklist is not possible given human cognitive limitations. Model-based methods provide an effective solution to this problem. System information (e.g., subsystem interrelationships, tailored views) can be implemented in models in a compact and manageable form. As important, models can facilitate collaboration among individuals with expertise in different disciplines. Finally, gaps uncovered in the model can be used to focus collaboration with the express purpose of filling the gaps to the extent possible.

Certain aspects of MBSE are still in the nascent stages. The primary focus, thus far, has been on concept engineering, modest size applications, and system modeling language and tools [2, 7, 27, 28]. Methods for verification and validation (V&V) and test and evaluation (T&E) are still in the early stages. Even so, MBSE has clear momentum. For example, INCOSE has a vibrant working group that continues to grow. IEEE Systems, Man, and Cybernetics Society (SMCS) has set up a MBSE Technical Committee that is focused on MBSE advances and is seeking to actively collaborate with the INCOSE Working Group. New case studies are being undertaken by the INCOSE MBSE Working Group to get a better handle on modeling with SysML while uncovering semantic gaps. MBSE courses are now required by aerospace prime contractors (e.g., Lockheed Martin, Boeing), and MBSE courses have become part of the Graduate M.S. Programs at major universities (e.g., USC, Georgia Tech, GMU, Johns Hopkins, Stevens Institute, MST). MBSE has become a best practice in SE at major national laboratories (e.g., JPL, APL).

Today, the focus is on specific SE questions that MBSE needs to answer. The inability to answer these questions indicates methodological deficiencies, and/or semantic gaps, and/or wrong level of abstraction in system representation. Representative questions that MBSE is intended to answer include: can a system model help evaluate a tradespace; what models are needed; what questions should the model address; when is the model considered to be complete; can physics models be made to correctly interface with architectural models; and how will models assure consistent semantics.

Some of the advantages MBSE has over current SE practices are: transparent, traceable design; reusable, metadata-tagged design fragments/components; libraries of standard and custom models; discipline-specific views for different types of engineers (e.g., MEs); automated model configuration management; support for different methodologies and life cycle models; multiple methods for requirements characterization (models, prose, graphics); support for virtual enterprise/supply chain integration; model-based verification and validation (V&V), model-based test and evaluation (T&E); reduced costs, higher quality, reduced time-to-market; and consistent and unambiguous communication among stakeholders.

22.3 Role of Ontologies and Meta-models in MBSE

Ontologies and meta-models are an integral aspect of MBSE. Ontologies are formal, explicit specifications of shared conceptualizations for specific domains. They define key terms and relationships in system models [18, 19]. Ontologies are constructed for: developing and sharing common understanding of the structure of information among human and software agents; enabling reuse of domain knowledge; making domain assumptions explicit; maintaining separation of domain knowledge and operational knowledge; analyzing domain knowledge; capturing agreement on usage; and enabling consistent (not necessarily complete) conversation and thereby preventing confusion and misunderstanding. Ontologies define a

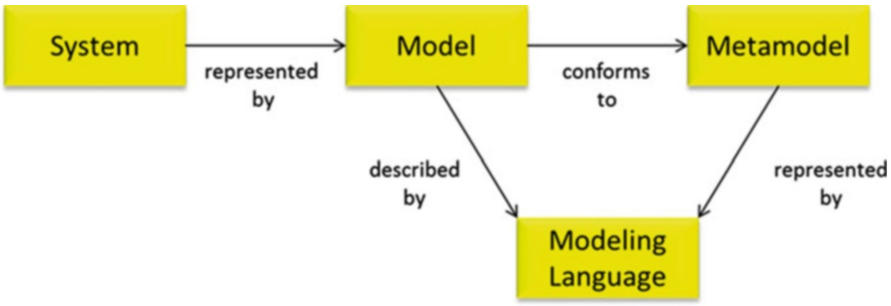


Fig. 22.3 Key relationship diagram

common vocabulary for a particular domain and the relationship between those concepts. Ontologies can be used to create domain models and support reasoning within the domains. Ontologies also facilitate communication among collaborators and different disciplines. They help circumvent problems that typically arise from the use of inconsistent terminology, data, interfaces, and assumptions. In MBSE, model elements can be traced from an abstract model to progressively more specific (concrete) models, and vice versa. This capability is achieved through meta-modeling and meta-models. A meta-model is the formal definition of the properties of a model, that is, a model that specifies the abstract syntax used by a modeling language. A meta-model is also a representation of a class of models expressed in a language. All meta-models are ontologies but not all ontologies are specified as meta-models. The relationships among a system, a model, a meta-model, and a modeling language are presented in Fig. 22.3.

Ontologies are also related to the concept of semantic modeling, which focuses on the meaning of entities and the relationships between them (Fig. 22.4). A semantic domain is a conceptualization (i.e., abstraction or model) of the real world defined by closed-world concepts and rules that govern the relationships between concepts ("rational world").

22.4 Role of Modeling and Simulation in MBSE

Modeling and Simulation (M&S) is a computational approach for developing an understanding of the interactions among components of a system, and the system as a whole. As such, M&S is at the heart of MBSE. The depth and breadth of models and simulations depend on the available data and reasonableness of assumptions made in the absence of data. With respect to MBSE, M&S is used for a variety of purposes including, but not limited to, trade-offs analysis, sensitivity analyses, what-if exploration of function allocation options, human–systems integration, system design and concept engineering, verification and validation, test and evaluation, and education and training.

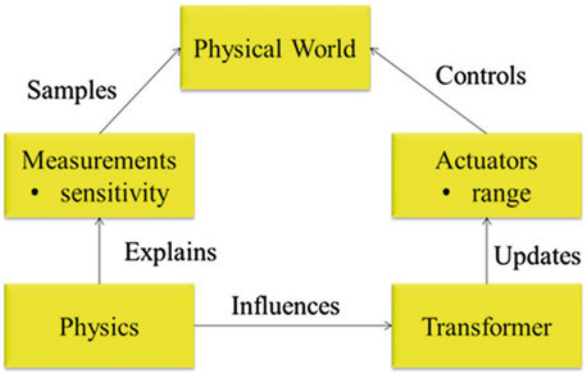


Fig. 22.4 Exemplar semantic model

The motivation for M&S stems from the recognition that humans are constrained by linear thinking in a world that is nonlinear. What this means is that it is next to impossible to understand how the various parts of a system interact and add up to the whole. It is equally impossible to explore all possible future behaviors of complex systems even with computer aids. As important, it is virtually impossible to foresee the full impact of cascading effects/events with limited mental models.

In light of the foregoing, the primary purpose of M&S is to explore the problem space and gain insights. Examples of insights are uncovering trends, cause–effect relationships, and correlations. A responsible attitude to M&S is that M&S provides a basis for constructing reasoned arguments about why certain outcomes are more or less likely than others.

Despite the common use of the term "M&S," the relationship between modeling and simulation is often not well-understood. From our perspective, simulation is a purposeful manipulation of a system model using appropriate data to answer what-if questions about the behavior of a model. Simulation enables controlled evaluation of the model. It makes the behavior of the modeled system apparent over time and space by allowing the experimenter to slow down or speed up simulation time. In other words, a simulation can run in real-time, faster-than-real-time, or slower-than-real-time. Simulation is usually a more cost-effective and faster alternative to real-world experimentation. It offers greater flexibility to evaluate modeled system behavior for various combinations of factors as well as enabling investigation of behaviors that are not easily observed in physical systems. A simulation can be deterministic or stochastic, discrete or continuous, single shot or Monte Carlo, and representative case or worst case.

22.5 From Traditional SE to MBSE: Key Challenges

A key SE challenge is achieving effective communication within and among stakeholders, that is, the individuals and organizations involved in specifying, using, maintaining, deploying, designing, and testing the system. A collaborative SE team needs certain information in common to establish a shared context for discussion. Such information typically includes key system requirements, business/mission/operational context, usage scenarios, key external interfaces (to other systems and people), high-level architecture, and key technical performance measures. In large organizations, having a shared context is especially important for meaningful collaboration. These are some of the concerns addressed by MBSE.

Another problem in SE is the frequent use of informal block diagrams to communicate within teams and across stakeholder groups. Block diagrams employed as a communication tool by systems engineers tend to be imprecise with ambiguous semantics that can quickly become a source of confusion. Yet another problem is the lack of rigor when analyzing needs. It is important to realize that needs come from stakeholders with diverse backgrounds, languages, and expertise. Stakeholder needs tend to often conflict and, therefore, require ongoing negotiation on their part. The challenge is determining how best to represent stakeholder needs so that their intent is clear to all stakeholders. Having such a representation of needs is a prerequisite to meaningful discussions about the relative importance and merits of various needs. The problem here is that stakeholders seldom share a common vocabulary to express and explain their needs and wants. Not surprisingly, they employ informal approaches for representing needs. These approaches invariably take the form of text documents along with an assortment of block diagrams. The latter typically have incompatible and inconsistent semantics. As a result, it is not possible to check consistency or ensure unambiguous statement of needs. These are some of the challenges that MBSE is beginning to address.

The biggest challenge to wide-scale MBSE adoption is gaining acceptance in the SE, program management, and acquisition communities, because MBSE does not readily fit the traditional documentation and review process that most customers and development organizations are accustomed to. While some progress has been made by tool vendors in producing documents and review packages from models, there is still a palpable gap between documentation produced from models and what customers expect. While training can be expected to eventually reduce the reliance on traditional SE artifacts, a subset of these artifacts will always be needed. As important, mechanisms are needed for incorporating artifacts/objects that are maintained outside the MBSE database. For example, detailed design drawings, which are created using specialized tools, are generally maintained in a configuration-controlled external database. Accessing these drawings from within the model, as well as maintaining consistency of the model with these drawings inevitably leads to parallel systems that while individually maintained, have to be mutually consistent.

In light of the foregoing, a key MBSE issue is to specify a common language for defining stakeholder needs. Stakeholder needs can be captured in one or more "use cases." These use cases can replace ad hoc statements by imposing structure and enforcing a consistent terminology when expressing needs. A use case comprises known facts about a need such as: who is interested in the use case; what is the triggering event or start time of the use case; what is the concluding event or completion time of the use case; what are the nominal, alternate, and exceptional behaviors defined by a use case.

The fact that there are methodological gaps in MBSE is not surprising, given that MBSE is still evolving. However, some gaps are more fundamental than others. For example, determining what constitutes a complete set of models is a fundamental gap that, in fact, is beyond the purview of MBSE. Also, the specification of model uses and how to use models is an overarching concern for all model-based approaches. Then there are MBSE-specific issues. For example, is it possible to have a single, unified model? If not, how should different heterogeneous models communicate? How should different disciplines and attendant models interact with each other? What measures need to be taken to assure common assumptions and consistent semantics across different models from the different disciplines? How should quality attributes be incorporated in system models and how can the models be analyzed in terms of the degree to which they satisfy the quality attributes? What is the best way to capture knowledge, decisions, decision rationale, and expertise of world-class system engineers? Lastly, since a model is a shared, living representation of multiple domains of interest, how can a consistent "baseline" be established, and how should it be reviewed?

Complex systems, which typically comprise a large number of strongly interacting elements (i.e., agents, processes), pose a unique challenge to MBSE. To understand these interactions, requires nonlinear modeling methods, nonequilibrium system representation methods, and new tools. The behavior of complex systems tends to be highly sensitive to initial conditions and/or small perturbations (i.e., small variations in state get magnified over time). In complex systems, the number of interacting components tends to be large, and/or multiple pathways exist by which the system can evolve. Complex systems tend to exhibit emergent behavior, that is, behavior that is not explicitly encoded in agents and that is not a property of the components. Emergent behavior results from interaction of agents/components within the system, as well as their interaction with the external environment. Also, complex systems, including systems-of systems, tend to have uncertain or changing boundaries. These characteristics pose a modeling challenge in that the behavior of complex systems is state-dependent (i.e., system has memory of state), and complex systems often result from a dynamic composition of systems (e.g., system-of-systems). These characteristics (e.g., nonlinear behavior, multiple feedback loops, changing or uncertain boundaries) make complex systems difficult to validate, test, and evaluate.

22.6 Promising Research Directions to Achieve Needed Advances

MBSE is intended to overcome certain key deficiencies in traditional systems engineering approaches. These include inconsistencies arising from the use of heterogeneous models because of differences in assumptions and modeling semantics, and having to rely on inconsistent, out-of-date, disconnected documentation. In MBSE, a collection of integrated models represents the sole "source of truth." These integrated models, which are persistently stored in a repository and allowed to evolve, are used to answer increasingly more difficult and greater number of stakeholder questions. The purpose of models in MBSE is to facilitate understanding and deliver insights; support marketing; enable communication; support visualization and decision rationale documentation; enable verification and validation of system requirements, structure, and behavior; trace behavior to requirements; resolve discrepancies; and support performance analysis.

However, at the present time, MBSE is still in the nascent stages and does not currently support the full system life cycle. In light of the foregoing, there are several research directions that can be pursued to mature the MBSE approach. Two of the more promising directions are being able to reach a wider stakeholder community, and offering more comprehensive analytical capability. To reach a wider stakeholder community, MBSE needs to add visualization and experiential perspectives that nonengineering stakeholders can understand and provide timely and meaningful feedback. In this regard, there is ongoing research in developing experiential perspectives derived from technical storytelling to augment MBSE and experiential design languages [20]. The experiential perspective results from humans interacting with systems within stories that unfold in virtual worlds. Another key research advance that needs to occur is augmenting MBSE with descriptive and analytic models of humans. The research in this area is concerned with reflecting human capabilities and limitations in the human models employed in MBSE [25]. As importantly, MBSE needs to be able to support trade-offs analyses that encompass quality attributes such as adaptability, resilience, and security. Research in this area is quite nascent but holds great potential for advancing the field [23]. Also, model-based test and evaluation (T&E) is a fertile area of research. In fact, a MBSE T&E working group has been formed under the auspices of the International Council on Systems Engineering (INCOSE) to address this issue. In addition, recent advances in complex systems engineering methodologies need to be incorporated within the MBSE rubric. Specifically, complex systems modeling and dependency analysis methods that include matrix methods such as DSM [1, 5], ES-MDM [1], Change Propagation Analysis [1], and Dynamic ES-MDM [21] can be incorporated within MBSE.

The application of reduced order techniques to matrix methods can be used to simplify analyses and achieve both scalability and parsimony. Such techniques classically include clustering and reordering methods, which enable encapsulation of concepts into fewer row/column elements. Such techniques are useful for physical and task DSMs, especially the reordering of task DSMs. In addition, one can deduce correlations in relationships among matrix elements to suggest additional aggregations to further simplify the matrix representation. This can be done statically (which amounts to a clustering algorithm) or dynamically through simulations and analyses of the results. The latter can be achieved by integrating matrix representation and interactive storytelling approaches [21].

From an analysis point of view, reducing the matrix size has clear benefits. From a representation and cognitive perspective, model reduction potentially simplifies inspection. One caution in this regard, is that one may inadvertently end up hiding elements that may be contingently important (i.e., the correlation was perhaps contingent on a particular scenario/system story that may change). From a design point of view, the reduction in model order can be used to intentionally simplify the design (e.g., prescriptive insight from the clustering/ordering/correlation) in any of the ES-MDM domains (e.g., function, tasks, objects).

There also exists the potential for developing a translation scheme for representing systems in dynamic ES-MDM as part of a larger meta-framework for model-based systems representation and analysis. This approach would enable the investigation of a much larger array of systems considerations than would be possible within a single modeling framework (e.g., system dynamics, agent-based models) or graphical language (e.g., SysML, UML). The key idea would be to ground the larger framework in matrix representation and graph-based methods coupled with simulation/interactive system storytelling [20, 21].

Finally, MBSE needs to be more broadly defined as Model-Based Engineering (MBE) to address the cyber, physical, and social elements of the system. This broader perspective can make MBE the preferred approach for cyber–physical–social systems modeling, analysis, and design. A recent INCOSE MBSE workshop keynote [27] prognosticated that MBSE will advance first and fast along the “hard” (i.e., physics-based) engineering aspects and subsequently integrate with the “soft” (i.e., human, social, economic, environmental) aspects that influence systems engineering.

For MBSE, to fully deliver on its promise and gain widespread adoption, several organizational, methodological and developmental advances need to occur. First, management needs to enthusiastically get behind the cultural change implied by MBSE. The current culture of relying on document-centric systems engineering is arguably the most serious impediment to adopting MBSE in most organizations. Second, MBSE needs to reach out to a wider stakeholder community comprising both engineers and nonengineers. Third, MBSE methods need to be extended to cover the full system life cycle. Extending MBSE methods will require both methodological advances and development of supporting processes and tools. Fourth, MBSE needs to incorporate human behavioral models that can support the evaluation of joint human–system performance and illuminate contexts that give rise to human error and degraded human–system performance. Fifth, MBSE value proposition needs to be convincingly demonstrated on real-world problems. Specifically, the benefits of MBSE (e.g., elimination of rework, cycle time reduction, risk reduction, cost reduction) need to be shown to system acquisition managers, program managers, and systems engineers for real-world systems/SoS of interest. Sixth, recent advances in complex systems engineering methodologies need to be incorporated within the MBSE rubric. And last, but not the least, MBSE needs to show the value proposition in terms of impact to an organization's bottom line.

References

1. Bartolomei JE (2007) Qualitative knowledge construction for engineering systems: extending the design structure matrix methodology in scope and procedure. PhD dissertation, June
2. Bjorkman EA, Sarkani S, Mazzuchi TA (2012) Using model-based systems engineering as a framework for improving test and evaluation activities. Syst Eng. doi:10.1002/sys21241
3. Boehm B (1986) A spiral model of software development and enhancement. ACM SIGSOFT Softw Eng Notes 11(4):14–24
4. Box GEP, Draper NR (1987) Empirical model-building and response surfaces. Wiley, p 424. ISBN 0-471-81033-9
5. Eppinger SD, Browning TR (2012) Design structure matrix methods and applications (engineering systems). The MIT Press, Cambridge
6. Estefan JA (2007) Survey of model-based systems engineering (MBSE) methodologies. INCOSE MBSE Focus Group 25
7. Fisher GH Model-based systems engineering of automotive systems. In: Proceedings of 17th digital avionics systems conference, The AIAA/IEEE/SAE, vol 1
8. Fisher J (1998) Model-based systems engineering: a new paradigm. INCOSE Insight I(3):3–16
9. Jankowski D (1997) Computer-aided systems engineering methodology support and its effects on the output of structured analysis. Empir Softw Eng 2(1):11–38. Kluwer Academic Publications
10. Jorgensen RW (2011) Defining operational concepts using SysML: definition from the human perspective. In: Rockwell Collins I (ed) INCOSE International Symposium, Denver.
11. Karban R, Weilkiens T, Hauber R, Zamparelli M, Diekmann R, Hein A (2011) MBSE initiative – SE2 challenge team: cookbook for MBSE with SysML
12. Lambrecht MR, Ivens PL, Vandaile NJ, Miller JH (1998) Active nonlinear tests (ANTS) of complex simulation models. Manag Sci 44(6):820–830
13. London B (2011) A model-based systems engineering framework for concept development. MS thesis, MIT
14. Long D, Scott Z (2012) A primer for model-based systems engineering, 2nd edn. Vitech Corporation, Blacksburg
15. Madni AM (2014) Expanding stakeholder participation in upfront systems engineering through storytelling in virtual worlds, accepted for publication. Syst Eng
16. Madni AM (1988) HUMANE: a knowledge-based simulation environment for human-machine function allocation. In: Proceedings of the IEEE national aerospace & electronics conference, Dayton, May 1988
17. Madni AM, Balcerak R, Estrin G, Freedy A, Melkanoff M (1990) Computer-aided Concurrent Engineering (CACE) of infrared focal plane arrays: emerging directions and future prospects. In: Second national symposium on concurrent engineering, Morgantown, Wet Virginia, February
18. Madni AM, Lin W, Madni CC (2001) IDEON: an extensible ontology for designing, integrating and managing collaborative distributed enterprises. Syst Eng 4(1):35–48
19. Madni AM, Madni CC, Lin W (1998) IDEON/IPPD: an ontology for systems engineering process design and management (invited paper). In: Proceedings of the 1998 I.E. international conference on systems, man and cybernetics, San Diego, October 11–14, pp 2591–2596

20. Madni AM, Nance M, Richey M, Hubbard W, Hanneman L (2014) Toward an experiential design language: augmenting model-based systems engineering with technical storytelling in virtual worlds. In: Madni AM, et al. (eds) 2014 Conference on Systems Engineering Research (CSER 2014), Redondo Beach, March 21–22
21. Madni AM, Spraragen M, Madni CC, Ross A (2014) VisualAnalytixTM: identification and visualization of interactions and their consequences within complex systems, Intelligent Systems Technology Inc. Phase I final report, contract #W911QX-13LC-0082, January 16
22. Min BI, Kerzhner AA, Paredis CJJ (2011) Process integration and design optimization for model-based systems engineering. In: Proceedings of the ASME 2011 international design engineering technical conference & computers and information engineering conference
23. Neches R, Madni AM (2012) Towards affordably adaptable and effective systems. Syst Eng 15 (1)
24. Oliver DW, Kelliher TP, Keegan JGJ (1997) Engineering complex systems with models and objects. McGraw-Hill, New York
25. Orellana DW, Madni AM (2014) Human-system integration Ontology: enhancing model-based systems engineering to evaluate human-system performance. Proc Comput Sci:19–25
26. Paredis C (2011) Model-based systems engineering: a roadmap for academic research. Georgia Tech, Model-Based Systems Engineering Research Center
27. Stoewer H (2014) Model based systems engineering (MBSE): missing link in the digital enterprise strategy? In: Keynote Talk, INCOSE MBSE Workshop, January 26
28. Tepper NA Exploring the use of Model-Based Systems Engineering (MBSE) to develop system architectures in naval ship design. http://hdl.handle.net/1921.1/61910
29. Uckun S, Kurtoglu T, Bunus P, Turner I, Hoyle C, Musliner D (2011) Model-based systems engineering for the design and development of complex aerospace systems. SAE Aerotech Congress and Exposition 2011, October 18–21, Tolouse
30. Wymore AM (1993) Model-based systems engineering: an introduction to the mathematical theory of discrete systems and to the tricotyledon theory of system design, vol 3. CRC Press, Boca Raton

Chapter 23
High-Fidelity Simulation Surrogate Models for Systems Engineering

Alex Van der Velden

Abstract In this paper, we will present a method to approximate the behavior of high-fidelity simulation models with surrogates that are constructed from high-fidelity simulation results. High-fidelity N-code (FEA, CFD, logical) cosimulations can take as much as 1 h CPU-time for every real-time second of behavior prediction (Van der Velden et al. (2012) Probabilistic certificate of correctness for cyber physical systems. In: ASME 2012 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference, August 12–15, Chicago, DETC2012-70135). We propose to speed up this process by four orders of magnitude, so that it is fast enough to be used in real-time applications, interactive design, multidisciplinary optimization, and verification of cyber–physical systems. These surrogate models can then be wrapped as a Functional Mock-up Unit (Functional Mock-up Interface: http://www.functional-mockup-interface.org/index.html) and deployed in a system simulation model such as Modelica® as well as in embedded hardware.

The present method converts time-series field data (as a function of design variables) to nonlinear ordinary differential equations for behavior at specific sensor locations using arbitrary surrogate (or meta) models (e.g., radial basis functions). The flexibility to choose any type of surrogate modeling technique maximizes the chance a highly predictive model can be found. The nonlinear ordinary differential equations are then solved efficiently with a numerical integration scheme. We verified the present method by comparing the analytical solution of the near chaotic motion of a double pendulum for given initial conditions with those of a surrogate model created from Modelica samples from different initial conditions.

The present method complements two other approaches to accelerate systems simulation. In the case of parameter estimation, the differential equations are known, but some of equation constants are not. In the case of model reduction, the full physical model is known a priori, but the complexity of the model is reduced. In the present method, the important (in and external) state variables

A. Van der Velden (✉)
SIMULIA, Dassault Systemes, Johnston, USA
e-mail: alex.vandervelden@3ds.com

A.M. Madni et al. (eds.), *Disciplinary Convergence in Systems Engineering Research*, https://doi.org/10.1007/978-3-319-62217-0_23

need to be known, but not the model form. The approach is limited by ability of the user-selected surrogate modeling technique to capture complex interactions.

The first example shows the effect of the activation of a controller to suppress airfoil flutter. The surrogate model was based on time-series responses for a 4-code Abaqus™ standard FEA, Abaqus CFD, control-build and Dymola cosimulation for various controller frequencies. Each of these cosimulating models solves either partial differential equations or ordinary differential equations that are coupled by a cosimulation engine.

The second example shows the dynamic motion and internal forces of a full 3D automotive vehicle as a function of arbitrary road surface boundary conditions. The surrogate model was based on time-series of 80 states computed for a specific virtual test track by the Abaqus™ explicit solver that took 24 h on 32 CPUs to complete. Subsequently, the fast surrogate model was successfully used to predict structural durability for arbitrary road conditions.

Keywords Approximation • Surrogate model • Meta-model • FEA • CFD • Cosimulation • Systems • Flutter • NVH

23.1 Introduction

N-code high-fidelity (FEA, CFD, logical) cosimulations can take as much as 1 h CPU time for every real-time second of predicted behavior. An example of an N-code cosimulation is an active control system based on wing trailing edge flap deflected by a servo actuator controlled by an electronic control unit [1]. Here, the Abaqus™ /FEA code is used for the structural analysis, the Abaqus™ /CFD code is used for the aeroloads, while the actuators are modeled with Dymola™ and the digital controller is modeled with ControlBuild.™ Each software code has its own specialized solver (ODE, PDE), while a cosimulation engine controls the time integration between the solvers.

Even though the high-fidelity cosimulation can predict highly accurate and highly refined behavior, it is typically too expensive to be used during highly iterative model-based system design phase. In addition, it can be very hard (or nearly impossible) to test the correctness of the simulations (e.g., in terms of numerical convergence) to verify the states during benchmark tests. The cosimulation cost can be reduced proportional by reducing the most expensive simulation using a multi-abstraction approach [1]. This reduction in simulation time [1] is a *necessary* condition to simulation-based design optimization and design verification.

Behavioral research [19] shows that "experience" is gained through real-time experimentation. If action causes a direct result in a short time interval, then "experience" is gained. Actions that have long-term effects only result in "experience" for a small group of people. Likewise, if there is no interactivity (i.e., no opportunity to change the experiment/simulation) little experience is gained.

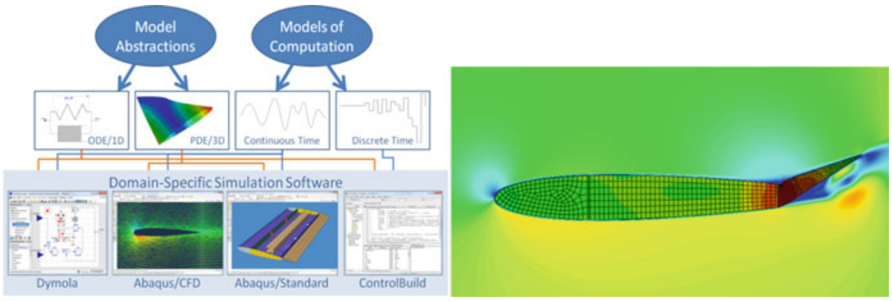


Fig. 23.1 N-code cosimulation of airfoil flutter (*left*). CFD flow field as predicted by Navier-Stokes for a partially separated flap during control system actuation (*right*) [1]

There are three existing approaches to reducing the computational burden and the solution complexity while maintaining solution accuracy over the domain of interest. The parameter estimation is probably the most widely used method. Pfeffer [7] defined the parameter estimation problem as follows: "Given a fixed structure for a dynamic model, an initial guess for the models parameters, and a set of frequency response data, find the parameters that make the model fit the frequency response data." The fitting operation is typically achieved by minimizing the errors between the frequency response data and the simulation data through optimization of the model parameters. A good example of the application of nonlinear parameter estimation on a problem similar to that of Fig. 23.1 is given by Klein [10]. A weakness of this approach is that the model form needs to be known a priori.

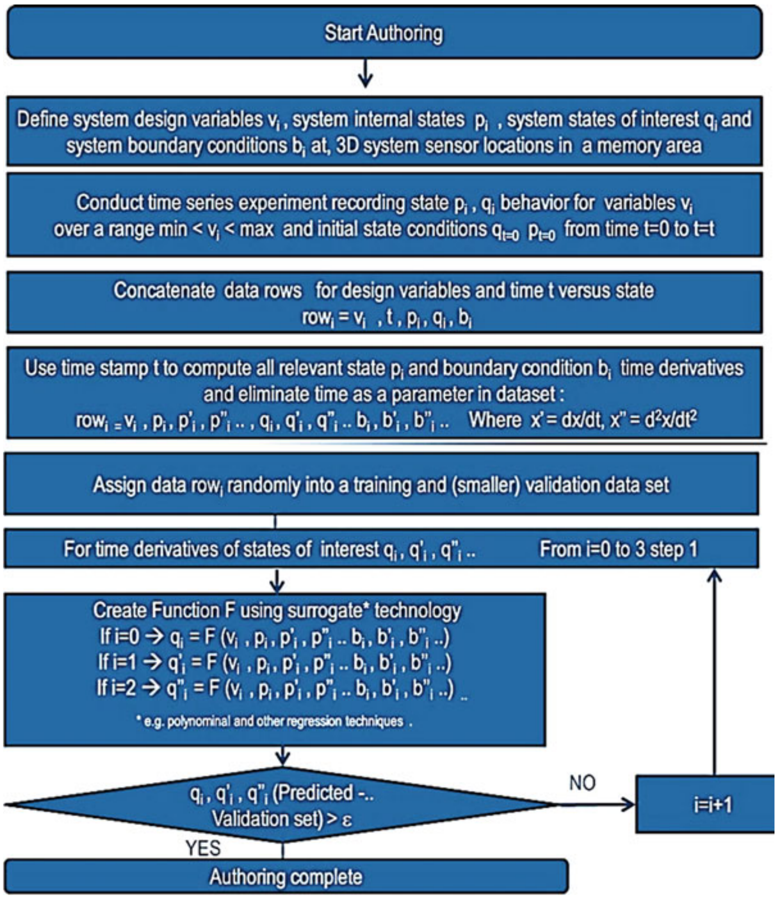
The second approach that is gaining interest is model reduction of nonlinear systems through the application of proper orthogonal decomposition by Karhunen [12] using Sirovich method [16] in combination with the trajectory piecewise linear [14] method to take into account nonlinearity. For chosen model states, linearizations of the reduced ODE model equations are stored along the solution trajectory. A benefit of this method is that it produces the same field information as the original high-fidelity simulation on which the reduced order model is based, whereas the parameter estimation method is limited to parametric data. However, this method does not allow for variations in the model as part of a scenario. In addition, the predicted values for individual sensor data (locations in the field) are not as accurate as they could be due to the inherent trade-off of minimizing the error for the entire field versus the error at discrete sensor locations.

However, if we are interested in creating fast running and accurate abstractions for design and verification, field data are typically not necessary. In model-based system engineering, behavior is only needed at a limited number of sensor locations. In addition, we want to leverage the utility of existing and widely used Design of Experiment (DOE) [15] and surrogate technology [13] to improve the capture of highly nonlinear systems and to study the impact of "design variables" in addition to dynamic states.

Such cosimulating surrogates can then be wrapped as a Functional Mock-up Units (FMUs) [2] and deployed in a system simulation tools such as well as in embedded hardware.

23.2 Present Method

Figure 23.2 shows a flowchart of the present method for authoring a surrogate for use in an interactive experience. The method begins by defining a simulation model representing a real-world system. The defined model includes system states and a design variable vector v. For example, the parametric state vector may include the position and angles of rotation (and their derivatives) of an object and the design variable vector may comprise the mass and the force (and its direction) on an object.



Start Authoring
Define system design variables v_i, system internal states p_i, system states of interest q_i and system boundary conditions b_i at, 3D system sensor locations in a memory area
Conduct time series experiment recording state p_i, q_i behavior for variables v_i over a range min < v_i < max and initial state conditions $q_{t=0}$ $p_{t=0}$ from time t=0 to t=t
Concatenate data rows for design variables and time t versus state $row_i = v_i, t, p_i, q_i, b_i$
Use time stamp t to compute all relevant state p_i and boundary condition b_i time derivatives and eliminate time as a parameter in dataset: $row_i = v_i, p_i, p'_i, p''_i .., q_i, q'_i, q''_i .. b_i, b'_i, b''_i ..$ Where $x' = dx/dt$, $x'' = d^2x/dt^2$
Assign data row_i randomly into a training and (smaller) validation data set
For time derivatives of states of interest $q_i, q'_i, q''_i ..$ From i=0 to 3 step 1
Create Function F using surrogate* technology
If i=0 → $q_i = F(v_i, p_i, p'_i, p''_i .. b_i, b'_i, b''_i ..)$
If i=1 → $q'_i = F(v_i, p_i, p'_i, p''_i .. b_i, b'_i, b''_i ..)$
If i=2 → $q''_i = F(v_i, p_i, p'_i, p''_i .. b_i, b'_i, b''_i ..)$ ..
* e.g. polynominal and other regression techniques.
q_i, q'_i, q''_i (Predicted -.. Validation set) > ε
NO
i=i+1
YES
Authoring complete

Fig. 23.2 Flowchart shows the present method of creating nonlinear ODE models for systems

The present method accelerates this realistic behavior modeling by first executing a physical or numerical high-fidelity experiment using a physical or numerical model and observing the response over time of the parametric state vector including internal states p and system states of interest q. This model can contain boundary conditions b such as discrete events (e.g., controller on/off) as well as externally forced conditions such as road excitations. Alternatively, before executing such an experiment, we may first reduce the dimensionality of the high-fidelity, high-dimensional numerical model using principal component analysis [6], or a similar technique [17], to a manageable number of parameters (e.g., approximately 100) before surrogation. This approach is repeated for instances of the design variable of interest v created by a Design of Experiment approach [15] to produce datasets as a function of (time, v, p, b, q) for a given set of initial conditions p(time = 0). The goal for the Design of Experiment technique is to produce evenly distributed samples in the hyperdimensional state and design space of interest. For the time-series, this is achieved in practice by using the initial state conditions as part of the DOE. However, it should be noted that this does not produce uniformly spaced training samples for the surrogate since trajectory traces are formed from each initial condition. Nevertheless, this approach is preferable to creating single (or excessively short) state time simulations, due to the simulation initialization cost. As such, there is a trade-off between the simulation initialization cost and the nonoptimality of long time-series simulations.

The method continues by concatenating the time-series datasets for each DOE and differentiating the variables with respect to time using a backward differentiation scheme and then removing time from the dataset, yielding (v, q,q',q"...,p,p', p'...', b,b'...). Then, the highest derivatives of the behavior of interest q is expressed a function of lower derivatives of q and the other vectors p, b, v and their derivatives. We can now leverage approximation techniques [13] such as radial basis functions, machine learning, Chebychev polynomials, and response surface methods with term reduction over the training set. The right predictive surrogate F is selected and its technique settings are optimized to reduce its variance and bias error and validated with the part of the dataset that is not used in the training.

The process is repeated for lower derivatives of the behavior of interest until an expression for F with the minimal sum of variance and bias error. When the error levels of the surrogate are similar for different derivatives of q, lower derivatives are preferred since they produce lower integration errors during execution.

Once the surrogates F are found for behavior q, it is now possible to numerical integrate q(t) from time = 0 to time = t for a given value v, given initial conditions p(time = 0) and variable boundary conditions b(t = t). These initial conditions of the state vector and design values are not limited to be the same as those of the original time-series experiments.

This model can in turn be exported to an FMU and be directly employed model-based systems to simulate realistic behavior in near real-time. Alternatively, it can be used directly in hardware, such as a flight simulator or an automotive electronic control unit (ECU).

23.2.1 Double Pendulum Model for Code Verification

The first example is principally for reproducible code verification. The equations of motion and their solution for the double pendulum can be found using Langrangian mechanics with generalized coordinates [18].

$$(m_1 + m_2)l_1\ddot{\theta} + m_2 l_2\ddot{\phi}\cos(\theta - \phi) + m_2 l_2\dot{\phi}^2\sin(\theta - \phi) + (m_1 + m_2)g\sin(\theta) = 0$$
$$l_2\ddot{\phi} + l_1\ddot{\theta}\cos(\theta - \phi) - l_1\dot{\theta}\sin(\theta - \phi) + g\sin(\phi) = 0$$

This simple model, as shown in Fig. 23.3, exhibits extremely complex nonlinear behavior that is highly dependent on the initial conditions. For relatively small angles (<0.5 rad), an analytical solution can be found [18] as a function of initial conditions and time.

We execute a Latin hypercube DOE where θ and ϕ are varied from -0.5 to 0.5 rad, and θ' and ϕ' are varied from -1 to 1 1 rad/s, whereas the l's are varied from 0.8 to 1.2. Figure 23.3 shows the double pendulum and its 500 DOE samples in the θ–ϕ plane. As a response, values of θ'' and ϕ'' are calculated.

We find that this sample can be approximated extremely well with Chebyshev polynominals in Isight™. Figure 23.3 (right) compares a trajectory based on the analytical solution to an approximated trajectory using a full fourth order (including cross-terms) Chebyhevs F_1 and F_2 each with 256 terms:

$$\mathrm{F}_1\left(\theta, \phi, \dot{\theta}, \dot{\phi}, l_1, l_2\right) = \ddot{\theta}$$
$$\mathrm{F}_1\left(\theta, \phi, \dot{\theta}, \dot{\phi}, l_1, l_2\right) = \ddot{\phi}$$

For large angles an analytical solution is not available. However, we can simulate the nonlinear equations of motion in Dymola™ and compare with the present method. We are interested in expanding the space to a solution where θ and ϕ can be varied from $-\pi$ to π and the l's can be varied from 0.1 to 3. The ranges of θ' and ϕ' are determined by the motion simulation. For this purpose, we generated 1000 DOE initial condition samples with 500 time-step time-series of the double

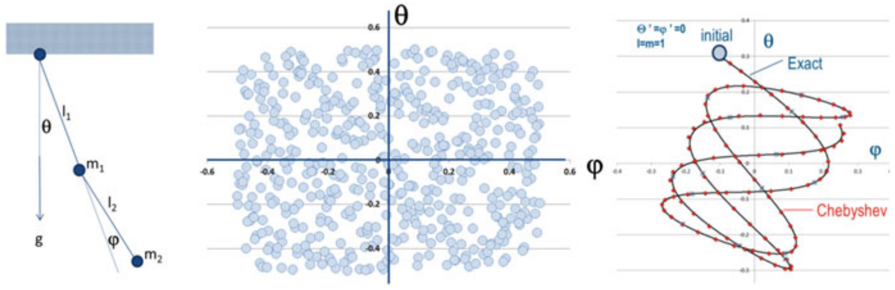


Fig. 23.3 Double Pendulum (*left*) and Latin hypercube samples (*center*) and analytical motion compared to present method

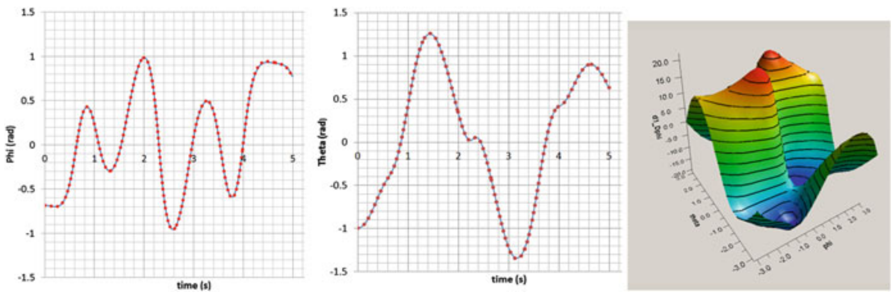


Fig. 23.4 Random validation trace of a large angle double pendulum simulation (*Red* = approximation, *Blue* = Dymola™) and partial depiction of state space F1

pendulum motion with Dymola™. This is more typical of the time-series samples we obtain from PDE-type simulations. However, this DOE produces a much less uniform distributed sample space, as the one shown in Fig. 23.3, and it is therefore much more challenging to reconstruct the state space from the samples. Also, very high accuracy is required for the forward integration and simple high polynomial models will fail on this problem. The problem was solved using a machine learning decision tree model with leaf regression. Figure 23.4 shows an example validation trace in the approximated space as well as a partial depiction of the state space. Note: The author can make this validation problem available to the reader for benchmarking.

23.2.2 Wing Flutter Control System

In this example, we start with a time-series produced by the nonlinear N-code simulation [1] described in the introduction. The cosimulation computed a 10 s (10,000 sample point) time-series in about 4 h. Figure 23.5 shows the cosimulation points in red. The data points are originally in the following form:

$$\mathrm{F}_1\left(\alpha, \dot{\alpha}, \mathrm{z}, \dot{\mathrm{z}}\right) = \ddot{\alpha}$$

$$\mathrm{F}_2\left(\alpha, \dot{\alpha}, \mathrm{z}, \dot{\mathrm{z}}\right) = \ddot{z}$$

where t = time, α= airfoil pitch, and z = airfoil vertical displacement (heave) on a wing section. In this first example, there is no controller or flap motion. The highest state derivative can be written as a function of the lower derivatives. Once we fit the 800 snapshot cosimulation points to a first-order response surface we create the following easy to understand linear ODE with 10 terms:

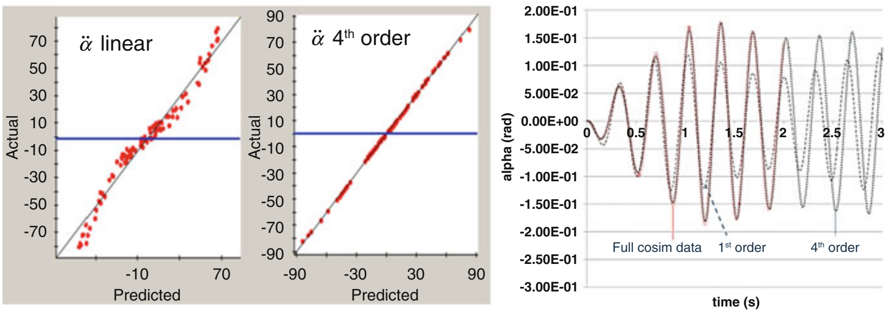


Fig. 23.5 Cross-Validation Error for the second time derivatives of α using a linear approximation (*left*) and a fourth-order approximation (*middle*). Time-series from Airfoil Flutter N-code simulation (*red*) approximated with a first- and fourth-order polynominal surrogate model (*right*)

$$\begin{bmatrix} \ddot{\alpha} \\ \ddot{z} \end{bmatrix} = \begin{bmatrix} -1.4 \\ 0 \end{bmatrix} + \begin{bmatrix} -423.11 & -127.73 \\ 18.513 & -230.74 \end{bmatrix} \begin{bmatrix} \alpha \\ z \end{bmatrix} + \begin{bmatrix} 2.5370 & 26.374 \\ 0.19421 & -3.2897 \end{bmatrix} \begin{bmatrix} \dot{\alpha} \\ \ddot{z} \end{bmatrix}$$

By looking at the cross-validation errors of Fig. 23.5, we can see a priori that $\ddot{\alpha}$ is not well approximated. However, if we increase the polynominal to fourth order with all second-order cross-terms, we can achieve cross-validation errors below 1% with just 23 terms. The fourth-order approximation is obviously necessary to capture the flow separation physics shown in Fig. 23.1. The z-values are accurate even for the linear approximation. Using the ODE approximation, we can now time-step the solution as shown in Fig. 23.5 (right). The linear ODE approximation obviously does not capture the physics very well as expected, while the fourth-order approximation is right on target.

We now add samples (for different initial conditions) with a 20, 25, 33, 50, and 100 Hz digital controller actuating the flap. Figure 23.6 shows a trajectory for different initial conditions and controller bandwidth (42 Hz) that were not covered by the original samples. Again, over time some state error builds up due to numerical integration (euler time-steps) and approximation, which leads to an offset between the time-series. When the controller is turned on at 3.1 s the full cosimulation creates a smooth transition to very high pitch accelerations $\ddot{\alpha}$ (-120 rad/s^2) due to flap deflection. The RBF surrogate jumps from controller off (0 Hz) to 42 Hz in a single time-step. From that point on, the acceleration predicted by the surrogate is smooth, whereas the full cosimulation with digital controller has high-frequency discrete corrections to the pitch acceleration.

23.2.3 Vehicle Road Excitation

For the final example we will apply the present method to motion of a full Abaqus™ 3D car model excited by a four-post rig to simulate road conditions. The variable

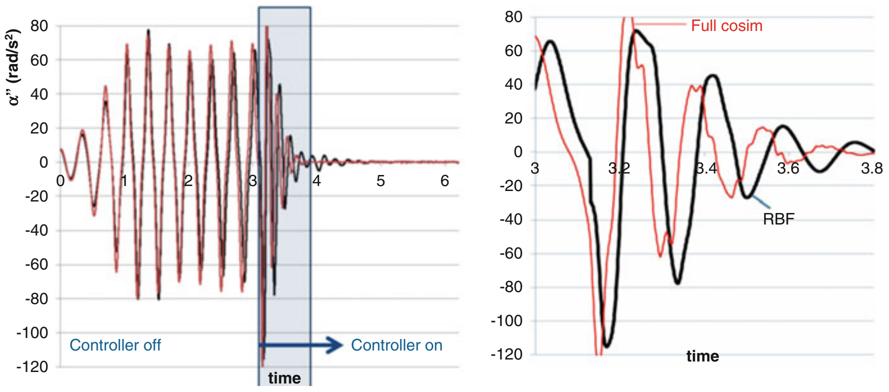


Fig. 23.6 Time-series from Airfoil Flutter N-code simulation (*red*) approximated with a fourth-order RSM with controller off and RBF with controller on at 3.1 s and 42 Hz. Figure 23.6 on the left on right represents the zoomed box of figure on *left*

road geometry is modeled as boundary conditions over time. Sensors, as shown in Fig. 23.7, are located on the 3D car model. The suspension system connects the wheels through the front rackhouse, so the wheels do not move independently on a bumpy road. The main conduit for the suspension forces to the vehicle structure are the damper tops. As shown below, the force at the top left damper is measured by a sensor in the simulation.

The challenge with this model is the very long simulation time. The body-in-white has a relatively inexpensive implicit FEA, but the realistic tire models are simulated with explicit FEA. A single 40 s simulation consists of 8000 time-steps, takes over 24 h on 32 CPUs, while recording over 80 states as a function of time. As a consequence, only one time-series simulation was made for this paper, which meant that we could not investigate the impact of design variables as we did in the previous example.

Figure 23.7 shows the trace of the rotational acceleration of the center of gravity as the car drives over different simulated road surfaces. The trace was divided into three pieces. First, the training piece of 10–30 s, next a secondary training piece from 30 to 40 s. The primary training time-series was used to find the approximation form. The secondary training piece was used to select its options by computing the error between the model based on the primary training set and the secondary training set data. The behavior in the first 10 s (as shown in Fig. 23.7) seems to be different from the rest of the time trace. We will use this time segment for validation.

This surrogate was then used to predict the sensor state time-series from 1 to 10 s on the basis of initial conditions on road boundary conditions alone. The author found that the response surface method (polynomial fit) with strong Efroymson term reduction [20] produced the best results on the secondary training set. The high number of input parameters resulted in over 1000 possible terms for a full second-order model. The R-values for this primary training set alone were typically 1.00,

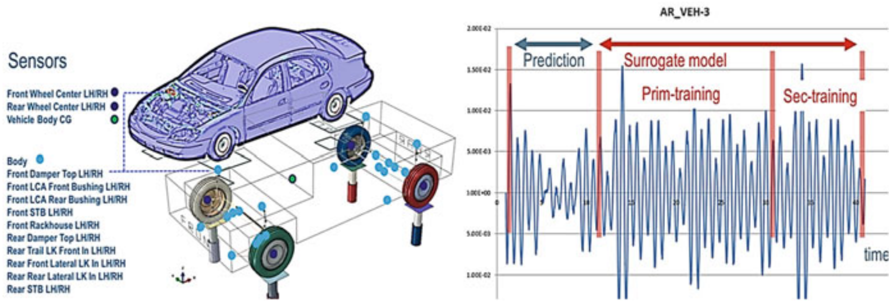


Fig. 23.7 Full Abaqus vehicle model with sensor locations. Dashed line is the location of the top front damper force sensor (*left*). Simulated time-series of the rotational acceleration of the center of gravity (*left*)

suggesting an excellent fit. However, the comparison to the secondary training set was typically poor due to variance error. This is probably because we do not have enough points in the primary training set. We were able to reduce the variance error by reducing the allowable maximum number of terms in the term selection. This provided significant (order of magnitude) additional benefits over the statistical method of [20] based on the primary training set alone.

It was found that first-order nonlinear ODEs best approximated the motion of the vehicle. The author is not entirely sure as to why this is, but it is probably because the damping forces dominate the problem and dropping some of the second-order effects improved the variance errors. Figure 23.8 shows the comparison of the model prediction versus the set-aside points from the simulation trace. There is drift due to cascading integration and approximation errors, but in general the validation is good.

The accuracy of this cosimulating surrogate model is good enough for durability studies and system suspension tuning. Figure 23.9 (left) shows the fatigue life of the body structure near the top damper due to the full FEA simulation and Fig. 23.9 (right) shows the fatigue life as determined cosimulating surrogate model of the top damper force. The full FEA model for the tire, suspension, and body is very expensive (1 h per simulated second) to simulate for arbitrary roads, whereas the reduced order model of the body FEA and the cosimulating surrogate model can be computed near real time (1 s per simulated second).

23.3 Conclusion

In this paper we propose a new method to create (co)simulation surrogates for highly nonlinear systems with active design variables that can be used during the model-based system design and verification phase. This method was designed to be

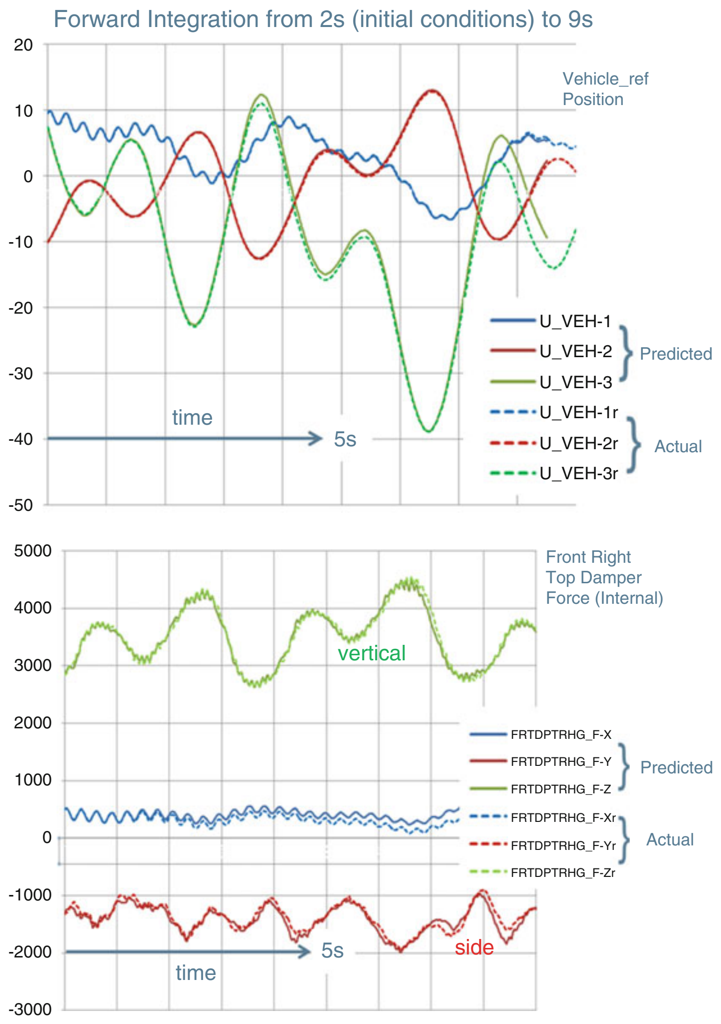


Fig. 23.8 Prediction of the vehicle motion (*top*) and vehicle internal damper force (*bottom*) for the simulation time-series of Fig. 23.7 from 2 to 9 s

broadly applicable to any simulation, but we focused on applying the method to mechanical simulations in this paper.

We used a combination of classical DOE [15] and Sirovich snapshot samples [16] to create nonlinear surrogates representing nonlinear ordinary differential equations for design variables and state variables. The ability to have choice [13] in the type of approximation seems critical to achieve the right approximation accuracy in the nonlinear region. Typically, the average error of the surrogate should not exceed 0.1%. to avoid instability in the numerical integration.

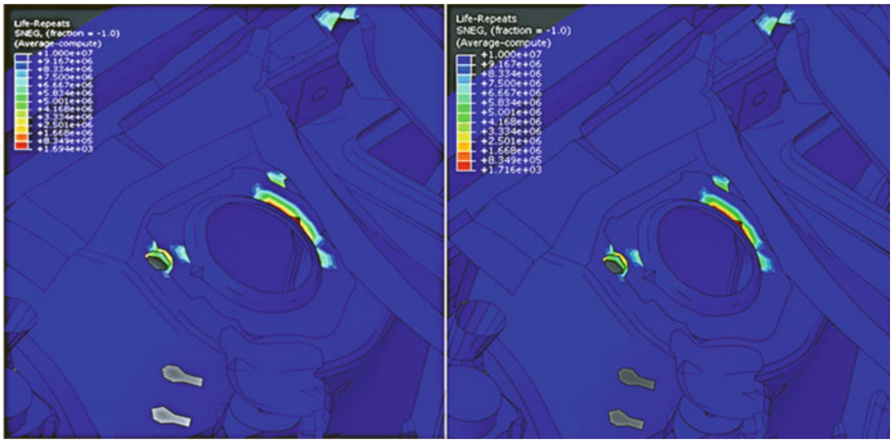


Fig. 23.9 Fatigue life of top damper structure using full FEA (*left*). Fatigue life of top damper structure using cosimulating surrogate model of top damper forces (*right*)

It should also be noted that sometimes we found that the inability to create a surrogate was due to the fact that the original simulation had model errors. Once these model errors were removed, we were able to successfully create a surrogate. As such, the surrogate approach can be an additional verification of the underlying data. This method was successfully applied on the practical example of a wing flutter control system and a vehicle NVH/durability model. In these cases, speedups of three orders of magnitude were achieved over the full-fidelity FEA models with little loss in predictive accuracy.

Looking forward, we expect that the present method surrogates can be combined with validated first principles models in order to fill gaps in modeling knowledge rather than using it to build the entire model.

Acknowledgments The author would like to thank Jeff Haan for generating the sample airfoil flutter sample data, as well as Fig. 23.1; and Youngwon Hahn for generating the sample data on the automotive NVH case, as well as providing Figs. 23.7 and 23.9.

References

1. Van der Velden A, Koch P, Devanathan S, Haan J, Naehring D, Fox D (2012) Probabilistic certificate of correctness for cyber physical systems. In: ASME 2012 international design engineering technical conferences & computers and information in engineering conference, August 12–15, Chicago, DETC2012-70135
2. Functional Mock-up Interface. http://www.functional-mockup-interface.org/index.html
3. Hardy RL (1971) Multiquadratic equations of topography and other irregular surfaces. J Geophys Res 76:1905–1915
4. Broomhead DS, Lowe D (1988) Multivariable functional interpolation and adaptive networks. Complex Syst 2:321–355

5. Kansa EJ (1999) Motivation for using radial basis functions to solve PDE's (unpublished: author kansa@IrisINTERNET.net)
6. Jolliffe IT (2002) Principal component analysis, series: Springer series in statistics, 2nd edn. Springer, New York
7. Pfeffer LE (1993) The RPM toolbox: a system for fitting linear model to frequency response data. In: 1993 Matlab Conference, October 18–20, Cambridge, MA
8. Dyson F (2004) Turning Points a meeting with Enrico Fermi. Nature 427:297. doi:10.1038/427297a
9. Adcock J (1987) Analyzer synthesizes frequency response of linear systems. Hewlett-Packard-J, January.
10. Klein V (1998) Aerodynamic parameters of high performance aircraft estimated from wind tunnel and flight test data, NASA-98-AGARD
11. Herkt S (2008) Model reduction of nonlinear problems in structural mechanics. PhD thesis, University Kaiserslautern. Heers B, Chin C, Kim M, Belsky V (2013) Computational advances in substructure generation and their implication of system-level analysis. NAFEMS World Congress
12. Berkooz G, Holmes P, Lumley JL (1996) Turbulence, coherent structures, dynamical systems and symmetry, Cambridge monographs on mechanics. Cambridge University Press, Cambridge
13. Devanathan S, Koch P (2011) Comparison of meta-modeling approaches for optimization. In: Proceedings of the ASME 2011 International Mechanical Engineering Congress IMECE2011-65541
14. Rewienski M (2003) A trajectory piecewise-linear approach to model order reduction of nonlinear dynamical systems. PhD thesis, MIT
15. Box GE, Hunter JS, Hunter WG (2005) Statistics for experimenters: design, innovation, and discovery, 2nd edn. Wiley. ISBN 0-471-71813-0. Taguchi, G (1995) Quality engineering (Taguchi methods) for the development of electronic circuit technology. IEEE Trans Reliab (IEEE Reliab Soc) 44(2):225–229. doi:10.1109/24.387375. ISSN 0018-9529
16. Sirovich L (1987) Turbulence and the dynamics of coherent structures I–III. Quart Appl Math 45(3):561–590
17. Jagenberg T (2008) Dimensional analysis in data approximation. Masters thesis, Institut for Flugzeugbau University of Stuttgart
18. Double Pendulum (2013) http://www.math24.net
19. Kahneman D (2011) Thinking fast and slow. fsgbooks
20. Efroymson MA (1960) Multiple regression analysis. In: Ralston A, Wilf HS (eds) Mathematical methods for digital computers. Wiley, New York

Chapter 24
Discovering Toxic Policies Using MBSE Constructs

Rahul Krishnan, Shamsnaz Virani, and Renato Gasoto

Abstract Policy development and implementation is considered an institution-centric endeavor. Most policies are documents that dictate institutional processes, organizational structure, and compliance standards. These policies are often developed by governing entities in isolation from the field. There is no standard method for evaluating a policy document and often toxic policies are realized during implementation, causing significant damage to the reputation of the organization. In this paper, we examine four Veterans Affairs (VA) polices using MBSE constructs of structural–behavioral integrity and consistency to investigate the toxic nature of policies. We also develop a framework to model policy, identify policy gaps, and calculate policy toxicity.

Keywords MBSE • Policy development • Model-based design • Policy analysis

24.1 Introduction

The Department of Veterans Affairs (VA) operates one of the largest integrated healthcare delivery systems in the USA, providing care to over 5.5 million veterans annually [1]. Owing to its large size, the policies that dictate department-wide procedures and operational requirements at the VA are increasing in number and complexity. The Government Accountability Office (GAO) recently conducted a study on the VA healthcare system and designated it as a high-risk area, with concerns about the VA's ability to ensure quality care of veterans [2]. It was found that ambiguous policy was one of the primary causes of low quality care, which led to inconsistent processes at the local level [2]. The increasing ambiguity in policy, concurrent with rising system complexity, is creating policy gaps, that is, failures in policy content.

Some government organizations like the Department of Defense (DOD) have policy writing guidelines that aim to reduce such inconsistencies and provide detailed information on document formatting, sentence structure, verb usage, etc.

R. Krishnan (✉) • S. Virani • R. Gasoto
Worcester Polytechnic Institute, 100 Institute Rd., Worcester, MA 01609, USA
e-mail: rkrishnan2@wpi.edu

[3]. However, such guidelines do not exist for the VA, which creates gaps in policy content. Some of these gaps were highlighted in the GAO study [2].

As of now, examination of policy gaps at the VA only takes place during or after policy implementation, allowing problems to creep into the system. The policy writing process at the VA is largely informal, with no emphasis on any structural or behavioral analysis of the policy. It is not possible to find all gaps in a policy based solely on the editorial review of its content. Evaluating and analyzing existing policy as well as establishing a formal policy writing process are key to identifying all gaps [4].

The field of systems engineering has a similar issue; the document-centric approach to systems engineering cannot cope with increasing system complexity. This gives rise to erroneous and inconsistent models [5], largely because a single systems engineer has to remember and trace every relationship present in the system. Gobet et al. showed that in most cases, a person cannot recall over three chunks of information at a time [6]. Likewise, Miller's law states that the number of objects an average human can hold in working memory is seven plus or minus two [7]. And here lies the problem: When analyzing a document, relationships between elements or components of that document are only present in the mind of the "analyzer" or "reviewer" [5]. In contrast, a Model-Based Systems Engineering (MBSE) approach overcomes this issue by providing traceability, consistency, robustness, and adaptability [5]. A set of structural and behavioral diagrams, describing all aspects of the system, implement such an approach. These diagrams are an essential construct of MBSE.

Clearly, adopting MBSE helps to better manage the complexity of systems. Likewise, we propose using MBSE to address the ambiguity and inconsistency that exist in policy while also handling increasing complexity. Utilizing MBSE constructs in the policy modeling process will help to evaluate and analyze policy content. In MBSE, these constructs are part of a process called Analytical Testing. An analysis such as this, before policy implementation, will help find gaps earlier (as seen from our results) and reduce risk to veterans' health and safety.

It is common to apply MBSE constructs to model business processes. Mattia Salnitri et al. proposed a framework for modeling and verifying the compliance of a business process model, that is, an activity diagram, with a set of security policies [8]. This is an example of a model-based approach, using Secure Business Process Modeling Notation-Query (SecBPMN-Q), an extension of Business Process Modeling Notation-Query (BPMN-Q) with security annotations. For verification, an algorithm checks if the business process model satisfies the security policies. However, this approach is not cross-domain and is only applicable on business process models [8]. If fewer well-defined processes and more requirements exist in a policy, building a comprehensive model of it using this approach will be difficult.

In this paper, we improve on the methodology proposed in Virani et al. [9] (to build a model of an existing policy) and introduce a novel method to calculate the "toxicity" or severity of existing gaps in a policy. We do so by using SysML as the modeling language to build behavioral and structural diagrams, that is, block definition diagrams, requirement diagrams, activity diagrams, and sequence diagrams. These diagrams offer a comprehensive model of the roles, requirements, and processes present in the policy. The “toxicity” value signifies (or shows) the number of gaps identified in the policy and its corresponding negative effect on the system. In the next section, we describe the methodology adopted and the process of building and analyzing policy.

24.2 Methodology

In this section, we introduce the methodology to policy modeling and analysis. We use the methodology proposed in Virani et al. as our base. It contains three distinct sections: policy selection, modeling strategy, and model checking. However, there is no provision in it for systematically identifying policy gaps. In our work, we extend the methodology in Virani et al. by adding a “gap investigation” section, which involves classifying, identifying, and analyzing gaps in the policy. We also add a step to the modeling strategy section, which is to select a “modeling construct.” Figure 24.1 illustrates the extended methodology. In the next paragraph, we will explain how we implemented this methodology on VA policies.

24.2.1 Policy Selection

As shown in Fig. 24.1, policy selection is the first step in the modeling process. In our case, this section was straightforward; Veterans Engineering Resource Center (VERC) gave us three policies to model. The key members of the modeling process were the research team at Worcester Polytechnic Institute (WPI) and members from VERC (who were also the Subject Matter Experts).

Here we focused our modeling efforts on Veterans Health Administration (VHA) policies relating to the Procurement and Logistics Office or the VHA P&LO. The policy dealt with VHA inventory management, logistics management, and standardization of equipment. These handbooks and directives provide operating procedures and requirements for personnel in their respective offices and departments. Handbook 1761.01 dealt with Standardization of Supplies and Equipment, Handbook 1761.02 was on VHA Inventory Management and Directive 7002, and Handbook 7002 was on Logistics Management Policy.

24.2.2 Modeling Strategy

We used SysML as the modeling language and a mix of manual and automation for the modeling process. In the manual process, the team reads a section of the policy, identifies the diagram that best describes it, and populates the diagram with its

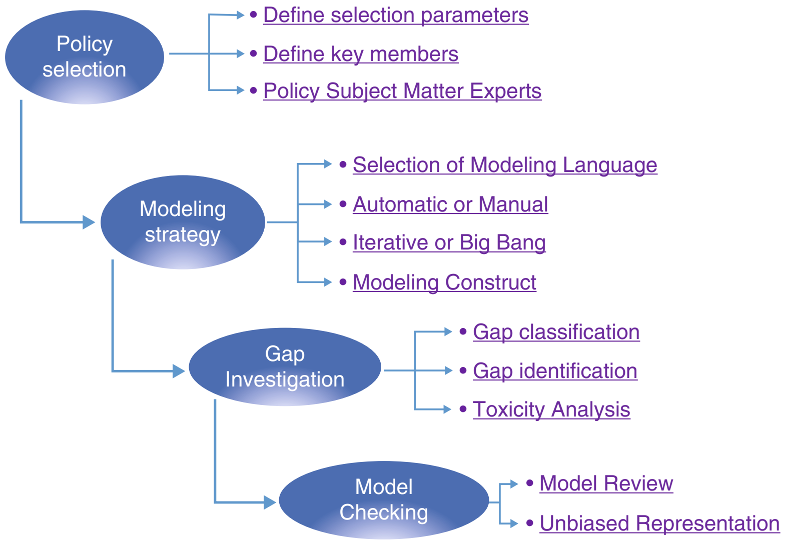


Fig. 24.1 Methodology

corresponding elements (SysML blocks – requirement, activity, role, etc.). This approach worked well for the smaller policies, that is, 1761.01 and 1761.02, each being 20 pages. However, the 7002 Handbook was a 150 page policy and adopting a manual process would have been time- and resource-intensive. Hence, an automated process was adopted for this policy. An automated process was favorable because the handbook only contained requirements and VERC staff had already classified sentences in the policy as requirements. Hence, developing an algorithm to convert text to requirement blocks helped automate the process and reduce lead time for modeling.

Selecting a modeling construct is another step added to the modeling strategy phase of the methodology in Virani et al. The goal is to choose between different constructs available to us based on the modeling language selected. In our case, structural and behavioral analysis was the modeling construct selected. In the next section, we will explain the modeling construct adopted by us in greater detail.

24.2.2.1 Modeling Construct: Structural and Behavioral Diagrams

The SysML diagrams we built model the structural and behavioral aspects of the policy. To develop the organizational structure of the policy, we divide it into different sections, following the same structure as the table of contents (to ensure an unbiased representation of the policy). A package diagram is ideal to develop the organizational structure, where a package represents a section of the policy. Each package contains one or more diagrams that further model that section. Figure 24.2 demonstrates the organizational structure of policy 1761.02.

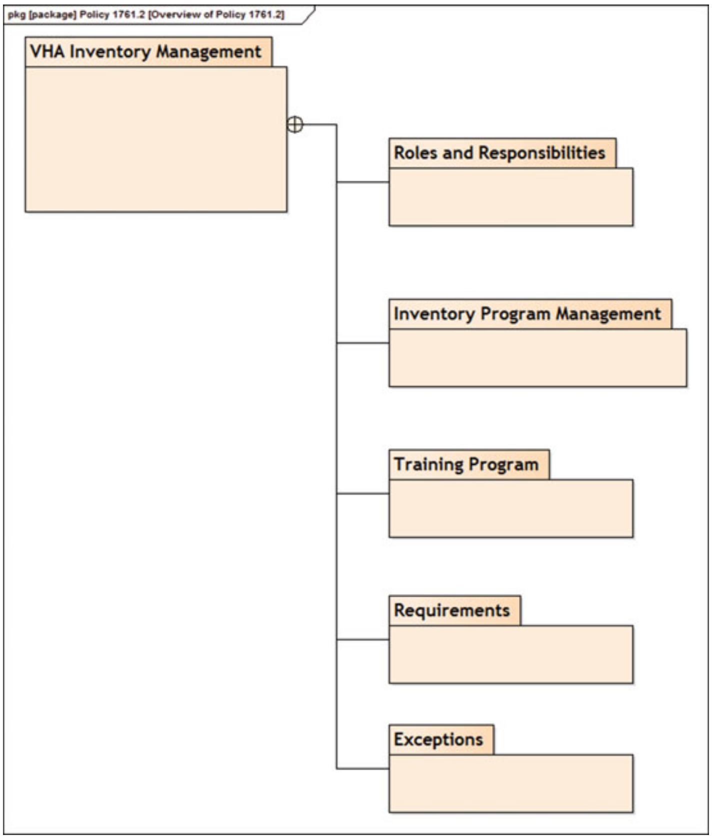

pkg [package] Policy 1761.2 [Overview of Policy 1761.2]
VHA Inventory Management
Roles and Responsibilities
Inventory Program Management
Training Program
Requirements
Exceptions

Fig. 24.2 Organizational structure of policy 1761.02 using a Package diagram

To identify the different actors in the policy and their associated responsibility, we build a Roles and Responsibilities diagram. A SysML "block" is used to model each role, with the actors' responsibilities listed in the block. Linking each responsibility to a requirement (in a requirements diagram) or an action (in an activity diagram) in the section or packages it exists in provides traceability to the model. Any transfer of information or documents between roles is shown using directional arrows, with a brief description of what is being transferred. Figure 24.3 shows a Roles and Responsibilities diagram for policy 1761.02. Finally, each block is also stereotyped as either Executive, Management, or Office, and color-coded to provide a visual aid (Fig. 24.4).

Since a majority of policies contain requirements, a Requirements Diagram is essential to model it . Figure 24.5 shows an example of a requirements diagram for policy 1761.02. Each requirement has a source that indicates its location in the policy document and helps trace it to its origin. Identifying relationships among requirements and other elements enable the modeler to define hierarchy and tag derived requirements.

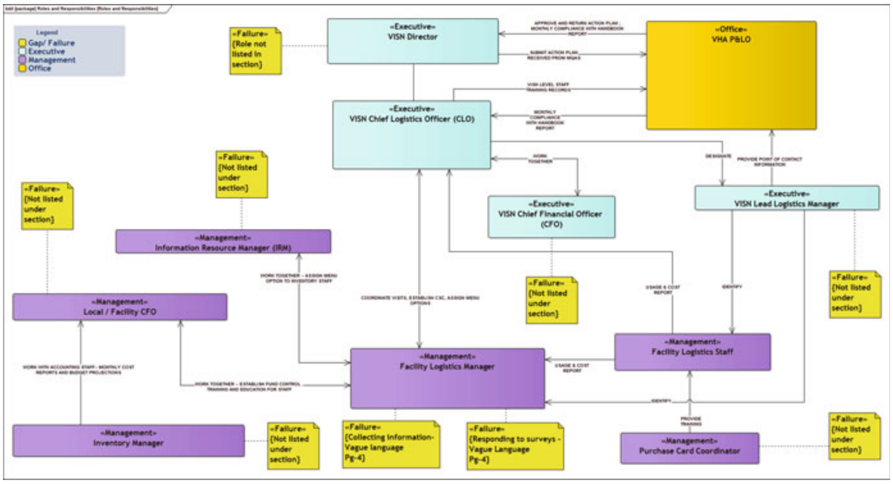
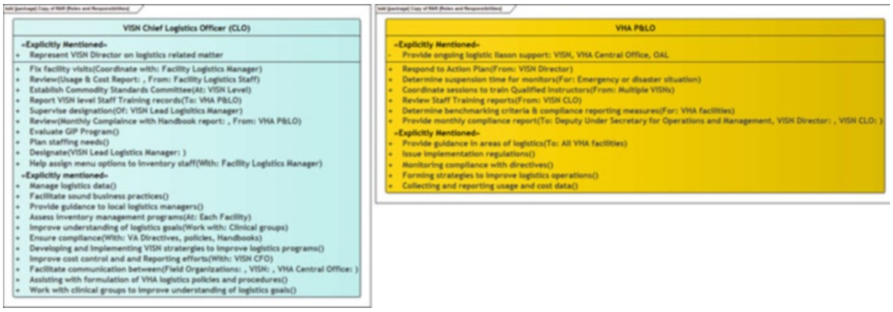

Legend
Gap/ Failure
Executive
Management
Office
«Executive» VISN Director
«Office» VHA P&LO
«Executive» VISN Chief Logistics Officer (CLO)
«Executive» VISN Chief Financial Officer (CFO)
«Executive» VISN Lead Logistics Manager
«Management» Information Resource Manager (IRM)
«Management» Local / Facility CFO
«Management» Facility Logistics Manager
«Management» Facility Logistics Staff
«Management» Inventory Manager
«Management» Purchase Card Coordinator
«Failure» {Role not listed in section}
«Failure» {Not listed under section}
«Failure» {Collecting Information- Vague language Pg-4}
«Failure» {Responding to surveys - Vague Language Pg-4}
APPROVE AND RETURN ACTION PLAN, MONTHLY COMPLIANCE WITH HANDBOOK REPORT
SUBMIT ACTION PLAN RECEIVED FROM MQAS
VISN LEVEL STAFF TRAINING RECORDS
MONTHLY COMPLIANCE WITH HANDBOOK REPORT
WORK TOGETHER
DESIGNATE
PROVIDE POINT OF CONTACT INFORMATION
WORK TOGETHER – ASSIGN MENU OPTION TO INVENTORY STAFF
COORDINATE VISITS, ESTABLISH CSC, ASSIGN MENU OPTIONS
USAGE & COST REPORT
IDENTIFY
WORK WITH ACCOUNTING STAFF - MONTHLY COST REPORTS AND BUDGET PROJECTIONS
WORK TOGETHER – ESTABLISH FUND CONTROL TRAINING AND EDUCATION FOR STAFF
PROVIDE TRAINING

Fig. 24.3 Roles and Responsibilities diagram



VISN Chief Logistics Officer (CLO)

«Explicitly Mentioned»
+ Represent VISN Director on logistics related matter

+ Fix facility visits(Coordinate with: Facility Logistics Manager)
+ Review(Usage & Cost Report: , From: Facility Logistics Staff)
+ Establish Commodity Standards Committee(At: VISN Level)
+ Report VISN level Staff Training records(To: VHA P&LO)
+ Supervise designation(Of: VISN Lead Logisitics Manager)
+ Review(Monthly Complaince with Handbook report: , From: VHA P&LO)
+ Evaluate GIP Program()
+ Plan staffing needs()
+ Designate(VISN Lead Logistics Manager:)
+ Help assign menu options to Inventory staff(With: Facility Logistics Manager)

«Explicitly mentioned»
+ Manage logistics data()
+ Facilitate sound business practices()
+ Provide guidance to local logistics managers()
+ Assess inventory management programs(At: Each Facility)
+ Improve understanding of logistics goals(Work with: Clinical groups)
+ Ensure compliance(With: VA Directives, policies, Handbooks)
+ Developing and Implementing VISN stratergies to Improve logistics programs()
+ Improve cost control and and Reporting efforts(With: VISN CFO)
+ Facilitate communication between(Field Organizations: , VISN: , VHA Central Office:)
+ Assisting with formulation of VHA logistics policies and procedures()
+ Work with clinical groups to improve understanding of logistics goals()

VHA P&LO

«Explicitly Mentioned»
- Provide ongoing logistic liason support: VISN, VHA Central Office, OAL

+ Respond to Action Plan(From: VISN Director)
+ Determine suspension time for monitors(For: Emergency or disaster situation)
+ Coordinate sessions to train Qualified Instructors(From: Multiple VISNs)
+ Review Staff Training reports(From: VISN CLO)
+ Determine benchmarking criteria & compliance reporting measures(For: VHA facilities)
+ Provide monthly compliance report(To: Deputy Under Secretary for Operations and Management, VISN Director: , VISN CLO:)

«Explicitly Mentioned»
+ Provide guidance in areas of logistics(To: All VHA facilities)
+ Issue implementation regulations()
+ Monitoring compliance with directives()
+ Forming strategies to improve logistics operations()
+ Collecting and reporting usage and cost data()

Fig. 24.4 Responsibilities of two roles listed in the Roles and Responsibilities diagram. Each block (or Role) in Fig. 24.3 has its own set of responsibilities. They can be viewed in the same diagram by adjusting the visibility settings in the software

To model processes in a policy, we use Activity diagrams. A process is broken down into a series of actions and decisions, with each action linked to the actor executing it. These diagrams help in understanding the flow of events in the process and make it easier to identify any gaps in sequences or missing actors for actions. Figure 24.6 illustrates an activity diagram for policy 1761.02.

After completing a diagram (requirement, activity, etc.), we rescan it element-wise, to find and tag gaps, if present. This is part of the Gap Investigation section of the methodology and is explained in the following.

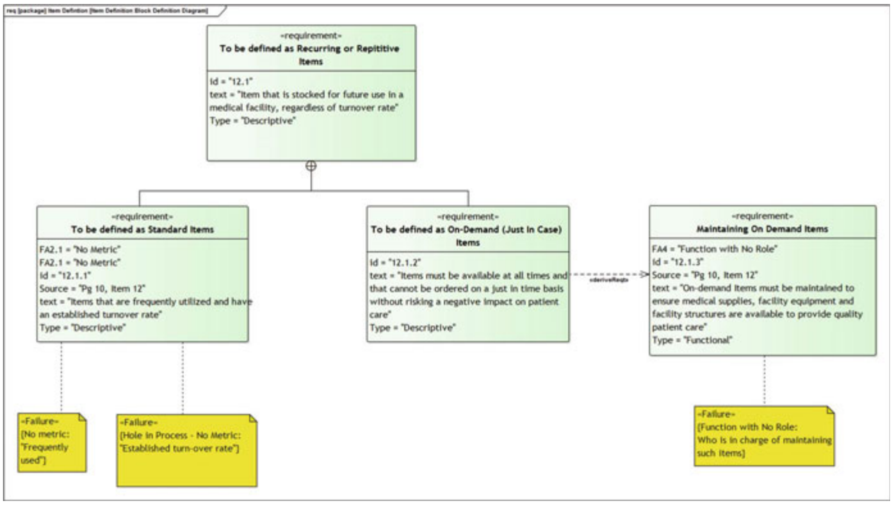

req [package] Item Definition [Item Definition Block Definition Diagram]
«requirement» To be defined as Recurring or Repititive Items
Id = "12.1"
text = "Item that is stocked for future use in a medical facility, regardless of turnover rate"
Type = "Descriptive"
«requirement» To be defined as Standard Items
FA2.1 = "No Metric"
FA2.1 = "No Metric"
Id = "12.1.1"
Source = "Pg 10, Item 12"
text = "Items that are frequently utilized and have an established turnover rate"
Type = "Descriptive"
«requirement» To be defined as On-Demand (Just In Case) Items
Id = "12.1.2"
text = "Items must be available at all times and that cannot be ordered on a just in time basis without risking a negative impact on patient care"
Type = "Descriptive"
«deriveReqt»
«requirement» Maintaining On Demand Items
FA4 = "Function with No Role"
Id = "12.1.3"
Source = "Pg 10, Item 12"
text = "On-demand items must be maintained to ensure medical supplies, facility equipment and facility structures are available to provide quality patient care"
Type = "Functional"
«Failure» [No metric: "Frequently used"]
«Failure» [Hole in Process - No Metric: "Established turn-over rate"]
«Failure» [Function with No Role: Who is in charge of maintaining such items]

Fig. 24.5 Requirements diagram for Policy 1761.02

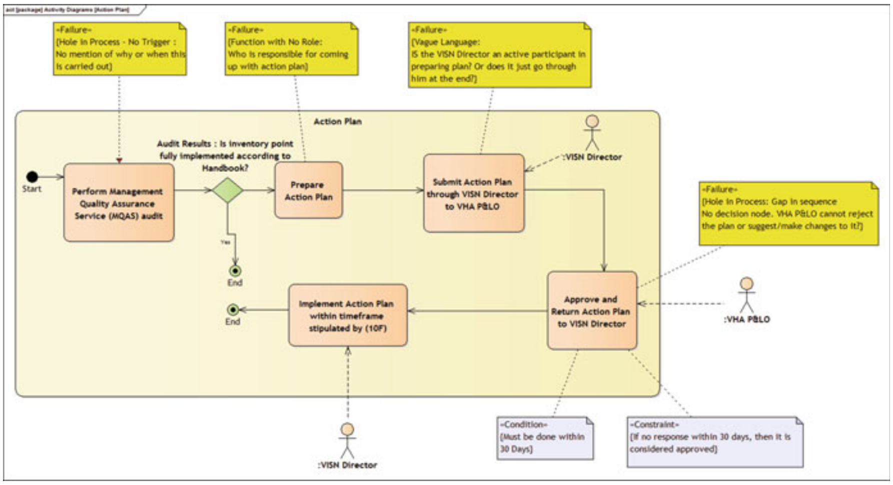

act [package] Activity Diagrams [Action Plan]
«Failure» [Hole in Process - No Trigger : No mention of why or when this is carried out]
«Failure» [Function with No Role: Who is responsible for coming up with action plan]
«Failure» [Vague Language: IS the VISN Director an active participant in preparing plan? Or does it just go through him at the end?]
Action Plan
Start
Perform Management Quality Assurance Service (MQAS) audit
Audit Results : Is inventory point fully implemented according to Handbook?
Yes
End
Prepare Action Plan
Submit Action Plan through VISN Director to VHA P&LO
:VISN Director
«Failure» [Hole in Process: Gap in sequence No decision node. VHA P&LO cannot reject the plan or suggest/make changes to it?]
Approve and Return Action Plan to VISN Director
:VHA P&LO
Implement Action Plan within timeframe stipulated by (10F)
End
:VISN Director
«Condition» [Must be done within 30 Days]
«Constraint» [If no response within 30 days, then it is considered approved]

Fig. 24.6 Activity diagram

24.2.3 *Gap Investigation*

Gap Investigation is the new section added to the methodology developed in Virani et al. The first step is Gap Classification, which involves creating a comprehensive list of failures applicable for the policy under consideration. Virani et al. developed a set of four gaps for policies related to healthcare [9]. While these gaps are not exhaustive, they can be applied to any policy. Using them as a base, with an emphasis on identifying gaps similar to the failures identified in the GAO report,

Table 24.1 List of gaps for Gap Classification

Gap Weight	Property	Gap Code	Gap Name	Structural	Behavioural
Medium	Incompleteness	1	Role with no function	Yes	Yes
Medium	Incompleteness	2	No Call Return	No	Yes
Medium	Incompleteness	2.1	- With no metric	Yes	Yes
Medium	Incompleteness	2.2	- With no verification	Yes	Yes
Medium	Incompleteness	3	Hole in process	No	Yes
Medium	Incompleteness	3.1	No trigger event	No	Yes
Medium	Incompleteness	3.2	Misplaced trigger	No	Yes
Medium	Incompleteness	3.3	Gaps in sequence	No	Yes
High	Inconsistent	3.4	Conflicting Target	Yes	Yes
High	Inconsistent	3.4.1	- Time	Yes	Yes
High	Inconsistent	3.4.2	- Role	Yes	Yes
Low	Inconsistent	3.4.3	- Requirement	Yes	Yes
Medium	Incompleteness	4	Functions with no role	No	Yes
Low	Verbosity	5	Unnecessary words	Yes	No
Low	Ambiguity	6	Vague language	Yes	No
Medium	Ambiguity	7	Undefined term	Yes	No
Low	Reference	8	Reference to other documents	Yes	No
Low	Verbosity	9	Redundancy	Yes	No
Low	Inconsistent	10	Misplaced information	Yes	No
Medium	Inconsistent	11	Role not Listed	Yes	No
Medium	Incompleteness	12	Requirement with no parent	Yes	Yes

we created a list of 11 gaps. Each gap is classified into one of five categories: incompleteness, inconsistency, ambiguity, verbosity, or reference. Table 24.1 shows the list of gaps used when modeling each of the four policies.

The next step is Gap Identification. As shown in Table 24.1, each gap has its own "code," which makes it easier to highlight gaps in the model. Hence, when an element in the model contains a gap, it is tagged with the code corresponding to that gap. The first column in Table 24.1 represents "Gap Weight," which distinguishes gaps that have a greater negative impact on the policy from those that have a lesser effect. Both, "Gap Weight" and "Gap Code" are crucial to the tagging process and are useful when running a qualitative analysis on the finished models. For example, if a requirement has a "Function with No Role" gap, that element is tagged as FA4. To improve visual identification of the gap in the model, a yellow box with a description of the gap is linked to the element. The modeler can add further text to help future modelers understand the reason for tagging. The advantage of such tagging is that most SysML software have the functionality to run UML scripts on the models, using which we can get a wide range of data on the gaps and the associated element. This analysis helps illustrate what the biggest concerns in the policy are and helps in calculating the "toxicity level" of the policy.

The final step in Gap Investigation is Toxicity Analysis. A toxicity level is the weighted average of all gaps present in the policy. As seen in Table 24.1, each gap is divided into one of three types based on Gap Weight – low, medium, and high, with a multiplication factor of 0.1, 0.5, and 1, respectively. The formula for finding the toxicity level of a policy is as follows:

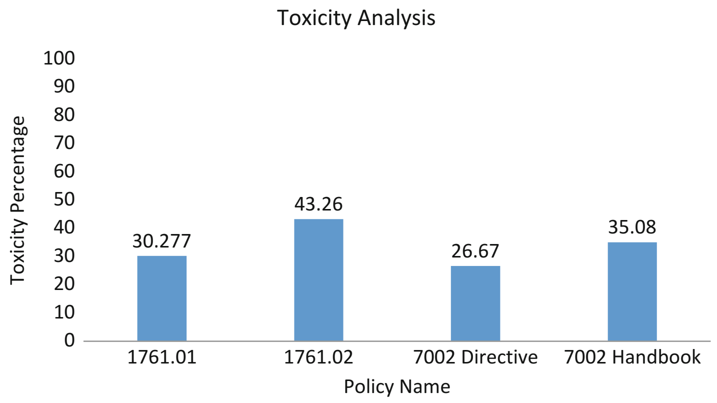


Fig. 24.7 Results from Toxicity Analysis of each policy

$$\text{Toxicity Level} = \frac{(0.1^* N_{\text{low}}) + (0.5^* N_{\text{med}}) + \left(1.0^* N_{\text{high}}\right)}{\text{Total number of failures}}$$

where N_{low}, N_{med}, and N_{high} indicate the number of low, medium, and high gaps, respectively.

The higher the toxicity level, the greater are the number of high-weighted gaps. Hence, a high toxicity level corresponds to a policy that has content written in a poor manner and can have serious consequences when implemented. Figure 24.7 shows the results from the Toxicity Analysis of the four VHA policies.

24.2.4 Model Checking

In the Model Checking phase, subject matter experts from VERC verify the finished model. They check if the elements used to model each section of the policy is appropriate (i.e., if the sentence in the policy is in fact a requirement or a process, etc.), check the validity of the gap identified for an element, and rectify any potential misinterpretation on the modelers' behalf.

24.3 Conclusion

This paper has introduced the use of MBSE constructs to model and analyze policy documents. The methodology we have developed can be used to model policy. Moreover, the concept of toxicity levels provides a measure for poor policy content.

Our approach overcomes the shortcomings of existing approaches, which are either lacking in functionality – due to its document-centric nature – or lacking broad use, being domain-specific. By applying a model-based construct to a policy document, we have not only identified gaps that an editorial review would miss but also determined levels of toxicity that can indicate future success or failure of the policy once implemented.

Acknowledgment We thank our colleagues from VERC New England Office who provided insight and expertise that greatly assisted the research, especially Dr. Tom Rust and Andrew Holbrook.

References

1. Williamson RB. (2011). VA Health Care : weaknesses in policies and oversight governing medical supplies and equipment pose risks to veterans' safety. Washington, DC: U.S. Govt. Accountability Office, May
2. Draper, D. A. (2015). VETERANS AFFAIRS HEALTH CARE: addition to GAO's high risk list and actions needed for removal. United States Government Accountability Office.
3. Department of Defense. (n.d.). DoD issuance standards. Retrieved from DoD Issuances The Official Department of Defense Website for DoD Issuances: http://www.dtic.mil/whs/directives/corres/writing/DoD_Issuances_Standards.pdf
4. Richardson S (n.d.) In using systems engineering tools to rethink U.S. Policy on North Korea. In: Perspectives on U.S. policy toward North Korea: stalemate or checkmate? Lexington Books, Lanham, MD
5. Harvey, D., Waite, M., Logan, P., & Liddy, T. (n.d.). "Document the model, Don't Model the Document *SETE APCOSE* 2012.
6. Gobet F, Clarkson G (2004) Chunks in expert memory: evidence for the magical number four ... Or is it two? Memory 12(6):732–774
7. Miller G (1995) The magical number seven, plus or minus two some limits on our capacity for processing information. Psychol Rev 101(2):343–352
8. Salnitri, M. (2014). Modeling and verifying security policies in business processes. In *Enterprise, Business-Process and Information Systems Modeling* (pp. 200–214). Springer.
9. Shamsnaz, V., & Rust, T. (2016). Using model based systems engineering in policy development: a thought paper. *Conference on Systems Engineering Research.*

Chapter 25
Model-Based Engineering: Analysis of Alternatives for Optical Satellite Observation

D.A. Shultz, J.M. Colombi, D.R. Jacques, and R.G. Cobb

Abstract This paper describes a research effort to employ modern Systems Modeling Language (SysML) tools to inform early architecture-level analysis of alternatives by integrating existing performance calculations through parametric diagrams. The purpose of this effort is twofold. First, it serves as a proving ground for the use of SysML to capture both the system architecture and the corresponding analysis framework. Second, it illustrates how to support specialty engineering performance calculations for a range of architectural options. The findings are illustrated using a collection of heterogeneous optical systems located at the Air Force Institute of Technology and used for observation of satellites. The model will include system elements decomposed to the level at which they are generally purchased, e.g., telescopes, cameras, telescope mounts, computers, etc. The range of options explored will be based on varying telescope aperture, sensor responsivity and noise, total system coverage, and background light as they affect systems performance. Metrics are examined based on calculation of the dimmest object detectable and time required to survey a band of sky containing geostationary satellites.

Keywords Model-based systems engineering • SySML • Analysis of alternatives • Tool integration

D.A. Shultz (✉)
Air Force Research Laboratory, Kirtland AFB, Albuquerque, NM, USA
e-mail: david.shultz.5@us.af.mil

J.M. Colombi • D.R. Jacques • R.G. Cobb
Air Force Institute of Technology, Wright-Patterson AFB, Wright-Patterson AFB, OH, USA
e-mail: john.colombi@afit.edu; david.jacques@afit.edu; richard.cobb@afit.edu

A.M. Madni et al. (eds.), *Disciplinary Convergence in Systems Engineering Research*, https://doi.org/10.1007/978-3-319-62217-0_25

25.1 Introduction

There are many system architecture modeling tools and many methodologies to develop system architectures [1]. Given this situation, it is no surprise that there are a great number of studies looking at the use of many combinations of tools and methods for a wide variety of systems. Rather than attempt to compile a representative list here, we will simply note that the tool vendors maintain lists of these references for the benefit of prospective and current users of their products. For the case of the Systems Modeling Language (SysML), the Object Management Group, Inc. (OMG) maintains the language standard, a sampling of books, papers, and links to vendor implementations of SysML tools [2].

Model-based Systems Engineering (MBSE) has seen computer aided design (CAD) tools increasingly exchange data with simulation tools. Using such tools at the detailed design phase develops a physical model, a parts list, and an executable model. Progress has not been so rapid for bridging the gaps between architecture-level tools and alternative system-level design tools; the gaps are even more pronounced for system-of-system architectures. A means to support conceptual and diverse architecture alternative solutions is needed. Further, it would be even better if the tools used to explore the alternatives could be extended to support later phases of systems engineering and could make use of existing tools and expertise.

In this paper, SysML is used to define a generic architecture for measuring satellite positions using optical observations on satellite targets in the geosynchronous region (aka GEO-belt). Beyond selection of the combination of optics and sensors, other significant factors include characteristics of the observing site to best meet users' goals. Section 25.3 will describe the process of building the architecture using the MagicDraw [3], Cameo Requirements Modeler, and SysML software environment, along with the ParaMagic [4] plug-in and MATLAB to calculate performance.

25.2 Background

25.2.1 *Systems Tools and Analysis*

Separate from the problem of the design tradespace is the problem of developing, and maintaining, an architecture in a form that can meet the functional and model exchange requirements of the user and the specialties involved in all phases of the system lifecycle. If the set of modeling tools is sufficient at the enterprise level, the model tools will support both the users' engineering and acquisition processes for all life-cycle phases from early architecture trades, through detailed design trades, and upgrades to fielded systems and, do so across the many engineering teams of the many companies engaged to support the system. It is clearly better to share a single model across the diverse teams at the various lifecycle stages, than to force the teams to rebuild separate models in a succession of Systems Engineering (SE) tools or default to putting data into static documents.

In addition to the system models discussed above, there are associated specialty calculations and supporting tools that can be quite disparate across the engineering teams, and it would be advantageous if these did not have to be rewritten multiple times. In the case of Teletrak, there were several scripts that had been written to perform particular calculations related to optical performance. Repurposing these preexisting scripts and turning some long scripts into a series of functions was relatively easy. The overall structure and order of the functions, based on inputs and desired outputs, was guided by architecture and, to a lesser extent, the capabilities and limitations of the MagicDraw and ParaMagic tools. Because the architecture was at a sufficiently high level, almost any tool or set of calculations could be substituted for the code used for Teletrak. This capability, to use existing specialty code to calculate dependent parameters based on inputs, is a key feature that enables model and code reuse in the performance of alternative analyses.

25.2.2 Model-Based Systems Engineering

While the Department of Defense (DoD) has the DoD Architecture Framework (DoDAF) and the DoDAF Metamodel (DM2) Physical Exchange Specification (PES) [5] standards for the data model and a data exchange format in the form of an XML schema, there is no design tool that natively uses this schema. Instead, each tool's internal data structures are translated to DoDAF views by unique code. Moving toward a more stable translation between tools, there is the Unified Profile for DoDAF and MODAF (UPDM) standard that also includes a standard for the interchange of machine readable models between different vendors' tools', but this only affects the subset of the model elements used in DoDAF/MODAF products. Work is ongoing to address current limits on users' ability to transition a complete model to another vendor's environment [6]. The DoD's, or any user's, motivation for interoperability is self-evident – disjointed SE activities, driven by different tool choices, result in disjointed, and inefficient, development and iteration of architectures, design products, specifications, test products, and integration information between teams.

Absent widespread portability of system models, document-centric practices continue to dominate DoD acquisition, not only in the acquisition-oriented framework, but also internal to individual program offices. This fact is one of the key issues surrounding system-of-system (SoS) engineering in the DoD; without detailed and interchangeable models among the interacting program offices, all of the exchange of technical details and performance modeling are themselves document based. In the cases where interactions are known and controlled early, such as in the Missile Warning domain, special effort is dedicated to interoperability and standards for data exchange in the form of requirements and collective testing

[7]. From outside the particular acquisition effort, it is impossible to say to what extent the exchange of models facilitates control of existing, or acquisition of future, component systems of a particular SoS. The potential advantages of unified model(s) to share among the stakeholders of the component systems of any SoS represent a clear benefit of MBSE.

25.2.3 AFIT Teletrak System

A brief description of Teletrak will be provided. For the purpose of this research, a system for satellite observation consists of the mount, (main) telescope optical tube, camera, and a computer of some sort to control these components. A secondary telescope and camera, separate focuser, filters, filter wheel, and accessory adapters and optics, may, or may not, be included depending on need. Defining the components in this way also aligns with their availability as discretely purchasable items. Further, a "GEO-belt object" is a phrase that includes geostationary, geosynchronous, near-geosynchronous, geotransfer orbits (GTO), disposal-orbit objects, and some Highly Elliptical Orbit (HEO) objects that exhibit motion that may be similar to those already listed during a given observation window. See Fig. 25.1.

The purpose of Teletrak was to serve as a platform for student projects in object tracking and collection of data for satellite orbit determination. The first task undertaken was development of mount guidance software to point and track the telescope at Low Earth Orbiting (LEO) satellites based on orbital elements. The initial demonstration was a single telescope, inexpensive webcam, and laptops running student-built tracking routines [8]. A succession of students continued to make improvements on automatic, closed-loop tracking [9], remote operations, and processing of observations for initial orbit estimation [10] and differential satellite orbits [11].

Today, the Teletrak suite of equipment has a much improved array of hardware and software. Telescopes are Meade Schmidt-Cassegrains of 10, 14, and 16 in. in aperture, generally used in tandem with 80 mm refractors as spotting scopes, and two Takahashi 106 mm wide field refractors. Imagers range from the original webcams to large format Charge Coupled Devices (CCDs) and one very sensitive back-illuminated CCD. Remote control capabilities have also evolved from merely pointing to include full camera control, focus, and filter wheel selection, with other capabilities in development to allow completely unattended operations.

While a single telescope (and associated sensor, computation, etc.) may not merit much of a system architecture, several scopes and sensors, each with their own strengths and weaknesses, used against a range of satellites, start to resemble the structure of the US Air Force Space Surveillance Network (SSN) of optical instruments. So, while Teletrak does not operate as part of the Air Force's SSN, it serves its purpose to educate students on the optics, network, and data processing elements of an SSN.

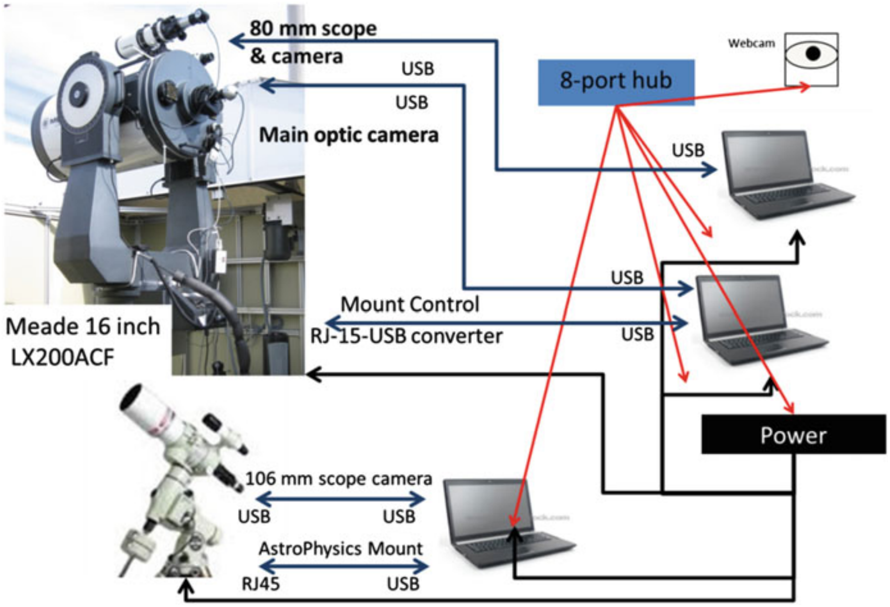


Fig. 25.1 Teletrak 16-in. scope in observatory

The decision was made to expand the optical system to support potential research out to geostationary satellites. The Air Force has been relying on optical systems for observations of these more distant satellites since the earliest days of the space age. To begin planning for the Teletrak upgrades, the design team reviewed the key assumptions and top-level objectives for the new system with the key faculty stakeholders:

- The entire system of scopes will be controllable from one location in Dayton, OH.
- The collection system will have sufficient capabilities to perform proof-of-concept research for determination of satellite positions.
- The collection system will be able to gather basic photometric measurements of objects.
- The upgrade has a constrained budget, so reuse of existing equipment is desired.

There are several considerations one can use to scope the problem of continuous surveillance of satellites and debris. First, only active operational satellites will change state, with the rare exceptions of satellites breaking up due to stored fuel or batteries exploding [12, 13]. Of the current 20,000 (approximate) tracked objects, only about 1000 are active [14]; thus, it is reasonable to sample the objects periodically. Second, most satellites are in Low Earth Orbits (LEO), up to roughly 1000 km altitude above the Earth. Only about 400 active satellites are in GEO-belt orbits [15]. Third, with the exception of the recent influx of CubeSats (all in LEO), most active satellites are fairly large, thus bright. Finally, a priority-based scheme can be adopted that may consider any number of conditions, but will always weigh whether the satellite is still actively maneuvered and how much time has passed since the last observations.

While most objects won't actively maneuver, the initial measurements and models for propagating the object's position forward in time are not perfect; there is a limit to how long one can wait to observe the object again before it becomes unlikely to be found where predicted [14]. Indeed, it is very problematic to maintain good orbits for some objects due to un-modeled, or even poorly understood, forces acting on them [16]. This provides for a challenging engineering solution on the design and modifications for the Teletrak system.

25.3 SysML Model of Teletrak

This research develops an architectural model of Teletrak that not only satisfies AFIT's immediate needs for a descriptive system model, but can be scaled up to capture the parallel structures of the SSN. In addition, while this effort was the first complete SysML model for Teletrak, several smaller calculation tools existed or were developed as part of the upgrade effort. It was one of the goals of this modeling effort to incorporate such purpose-built specialty engineering tools into the overall Teletrak SysML model. By building a SysML structure that calls on the specialty tools to provide parameters back into the SysML tool, the modeling effort will have achieved some of the objectives described in Sect. 25.2.1, and each tool, whether specialty calculation or SysML model-building, will be playing to their respective strengths to create a unified model of the system.

With respect to AoA-type analyses, by designing the domain model of the optical system with awareness of the specialty calculation tools (in the case of Teletrak, these are MATLAB functions) the model structure reflects the inputs and outputs of these tools. This situation, where specialty models have been previously built for related analysis, reflects a fairly common occurrence in the systems engineering of new systems that are interacting with users and preexisting models and simulations developed for earlier-generation systems. In the case of Teletrak, many generations and options exist for modeling optical performance paired with sensor performance. The addition of treatment for wavelength dependency through filters, background, and atmosphere extend these basic models. Lastly, as the models for all these elements were investigated before this system-level architecture was developed, decisions about level of detail and structure were made to align and reuse with the current physics-based models to allow their reuse.

The model began with a straightforward process of specifying SysML *BLOCKS* corresponding to the hierarchy of Teletrak system components. As shown in Fig. 25.2, the Teletrak model reused code for the optical and sensor elements (the SySML "merge" is used to highlight this reuse), but also includes a telescope mount, and the two types of workstation functions. Looking at the optical system,

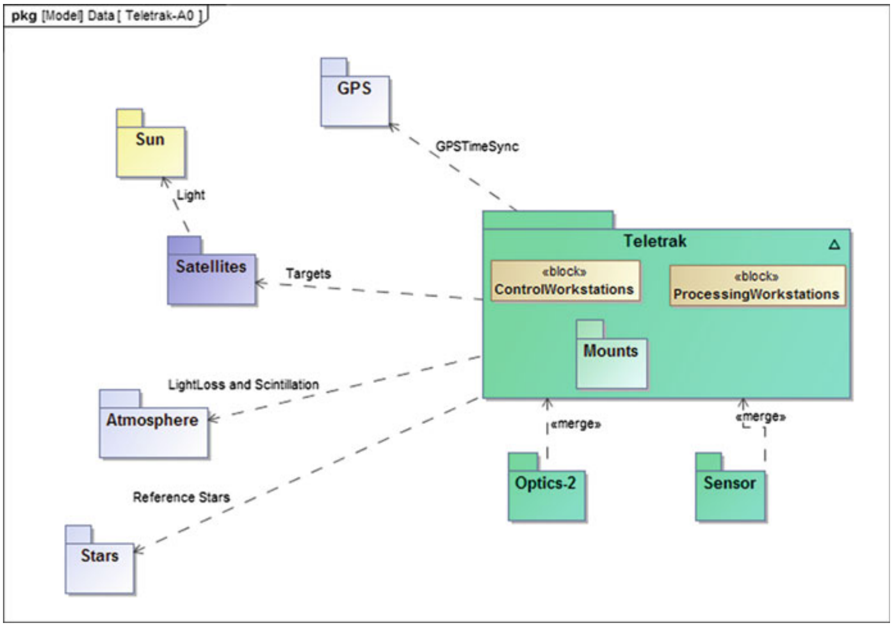

pkg [Model] Data [Teletrak-A0]
GPS
Sun
GPSTimeSync
Light
Satellites
Targets
Teletrak
«block» ControlWorkstations
«block» ProcessingWorkstations
Mounts
LightLoss and Scintillation
Atmosphere
«merge»
«merge»
Optics-2
Sensor
Reference Stars
Stars

Fig. 25.2 Package diagram for the Teletrak SoS

it made sense to break the components down to the level at which they can be purchased and upgraded, which produces a more structural viewpoint for any given assembled instrument (Fig. 25.3).

As the naming scheme is described, it is useful to keep the iterative cycle of the model-building in mind. Almost all entities have to reside in a *BLOCK*, so at each level of detail create the *BLOCKS* needed in the correct containment relationships. Within the BLOCKS, create *ValueProperties*, then move the BLOCKS to a block definition diagram (*bdd*) and use the *directed composition* relationship as required by the structure. Next, create the *CONSTRAINT BLOCKS*, each with one or more *constraint parameters*, and each parameter with its *constraint specification (property)*. This last step is the actual mathematical expression that operates on the input *ValueProperties* and returns the output *ValueProperties*. The last step is to create the parametric diagram whose structure will mirror the *bdd*.

Each "constraint' that needs to be modeled is actually a minimum of three objects in the model structure, not counting the similarly prolific names for the objects that will participate in each constraint calculation; this is the point where the need for a predefined naming scheme becomes obvious. A naming scheme should be planned so the modeler will not get lost navigating the model structure and it is clear which object should be chosen in the selection list menus.

For the Teletrak model, the decomposition described above was carried out for a few illustrative examples of the model objects. At this stage, all the *BLOCKS* shown in the package diagrams represent the general class of their type. The next step was

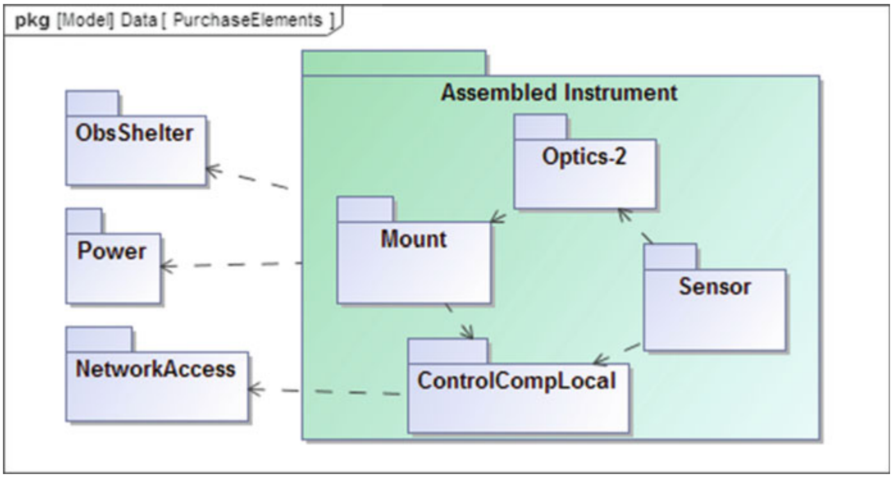


Fig. 25.3 Package diagram for an assembled instrument

to create the relationships, constraints, interfaces, etc. If an additional constraint is desired later, it will be necessary to modify the generic parent block then recreate the instance(s).

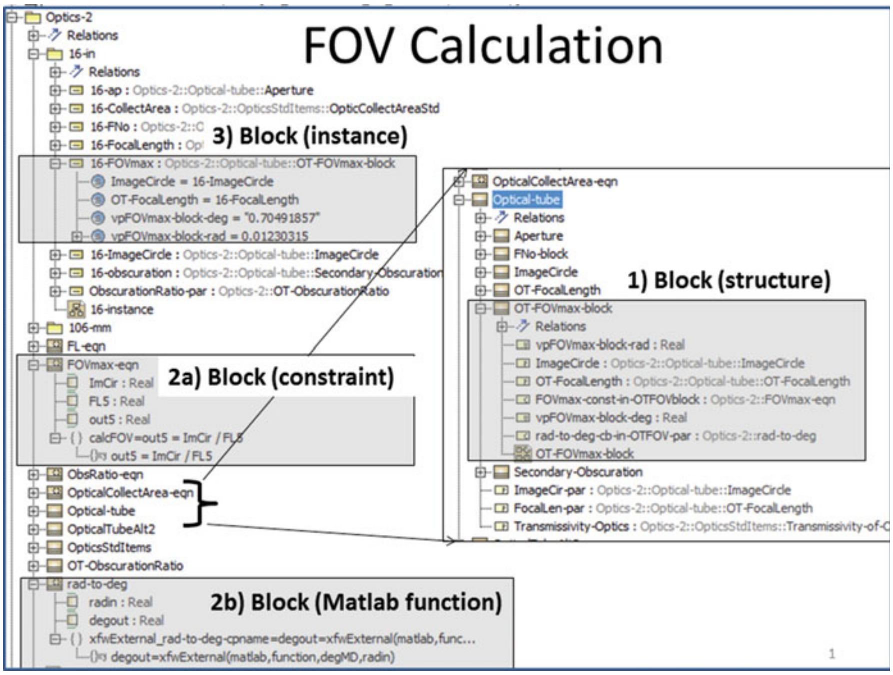
After the abstract *BLOCK* objects have been created, concrete instances of telescopes were created to portray the 16-in. Meade SCT, the 106 mm Takahashi refractor, and others. When actual numbers have been entered for the *slot property* of the *instance specification,* then the calculation can be carried out by the ParaMagic plug-in. The structure objects, highlighted in Fig. 25.4, correspond to the block and parametric diagrams for Field of View (FOV) calculation (Figs. 25.5 and 25.6).

A few items should be kept in mind while developing the naming convention. MagicDraw (as of version 18.1) does not allow underscores in object names. Since MATLAB does not take variable names with hyphens, the naming scheme for *Parameters* was limited to upper and lower-case letters; CamelCase names were used where lower case alone were hard to read. As for variable types, ParaMagic version 18.1 only supports 'real" data types for input or output. Clearly, MBSE tool integration is evolving, but there are still nuances in the detailed modeling and linkages.

25.4 Teletrak SysML Model Evaluation/Assessment

25.4.1 Evaluation/Assessment Process

In detail, the process of creating a calculation is as follows (see Fig. 25.7).



FOV Calculation

3) Block (instance)

```
Optics-2
  Relations
  16-in
    Relations
    16-ap : Optics-2::Optical-tube::Aperture
    16-CollectArea : Optics-2::OpticsStdItems::OpticCollectAreaStd
    16-FNo : Optics-2::C
    16-FocalLength : Op
    16-FOVmax : Optics-2::Optical-tube::OT-FOVmax-block
      ImageCircle = 16-ImageCircle
      OT-FocalLength = 16-FocalLength
      vpFOVmax-block-deg = "0.70491857"
      vpFOVmax-block-rad = 0.01230315
    16-ImageCircle : Optics-2::Optical-tube::ImageCircle
    16-obscuration : Optics-2::Optical-tube::Secondary-Obscuration
    ObscurationRatio-par : Optics-2::OT-ObscurationRatio
    16-instance
  106-mm
  FL-eqn
  FOVmax-eqn
    ImCir : Real
    FL5 : Real
    out5 : Real
    calcFOV=out5 = ImCir / FL5
      out5 = ImCir / FL5
  ObsRatio-eqn
  OpticalCollectArea-eqn
  Optical-tube
  OpticalTubeAlt2
  OpticsStdItems
  OT-ObscurationRatio
  rad-to-deg
    radin : Real
    degout : Real
    xfwExternal_rad-to-deg-cpname=degout=xfwExternal(matlab,func...
      degout=xfwExternal(matlab,function,degMD,radin)
```

2a) Block (constraint)

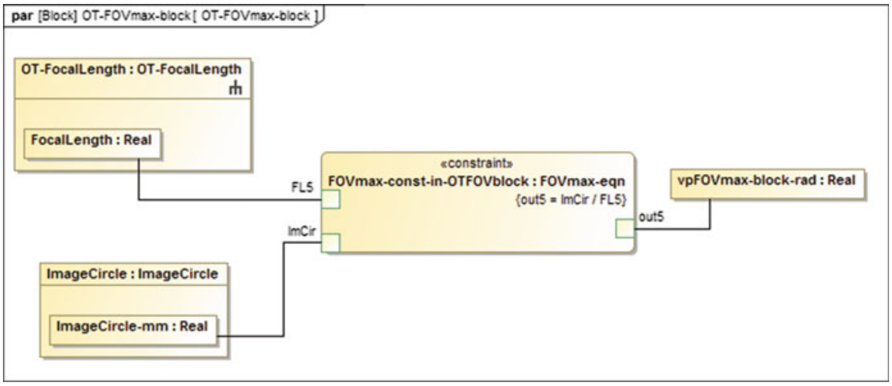
2b) Block (Matlab function)

1) Block (structure)

```
OpticalCollectArea-eqn
Optical-tube
  Relations
  Aperture
  FNo-block
  ImageCircle
  OT-FocalLength
  OT-FOVmax-block
    Relations
    vpFOVmax-block-rad : Real
    ImageCircle : Optics-2::Optical-tube::ImageCircle
    OT-FocalLength : Optics-2::Optical-tube::OT-FocalLength
    FOVmax-const-in-OTFOVblock : Optics-2::FOVmax-eqn
    vpFOVmax-block-deg : Real
    rad-to-deg-cb-in-OTFOV-par : Optics-2::rad-to-deg
    OT-FOVmax-block
  Secondary-Obscuration
  ImageCir-par : Optics-2::Optical-tube::ImageCircle
  FocalLen-par : Optics-2::Optical-tube::OT-FocalLength
  Transmissivity-Optics : Optics-2::OpticsStdItems::Transmissivity-of-C
```

1

Fig. 25.4 Model structure showing locations of BLOCKS created for FOV calculation



par [Block] OT-FOVmax-block [OT-FOVmax-block]
OT-FocalLength : OT-FocalLength
FocalLength : Real
FL5
ImCir
«constraint»
FOVmax-const-in-OTFOVblock : FOVmax-eqn
{out5 = ImCir / FL5}
out5
vpFOVmax-block-rad : Real
ImageCircle : ImageCircle
ImageCircle-mm : Real

Fig. 25.5 Block definition diagram of optical components created for FOV calculation

- Create (or reuse) the *BLOCKS* that will participate in the equation, to include the result(s). As you address each *BLOCKS*, create (or reuse) *ValueProperties* within them.
- Create, or modify, a *bdd* that includes the *BLOCKS* that will participate using the *Directed Composition* connector-type with directionality from result-to-input.

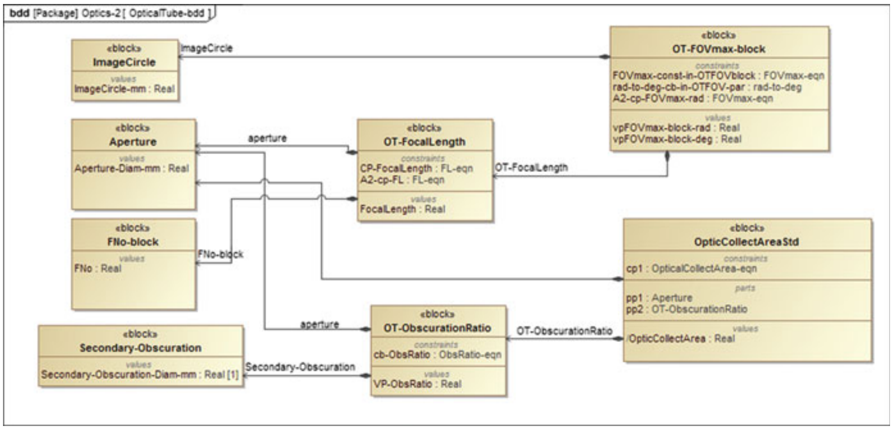


Fig. 25.6 Parametric diagram for calculation of optical FOV limit

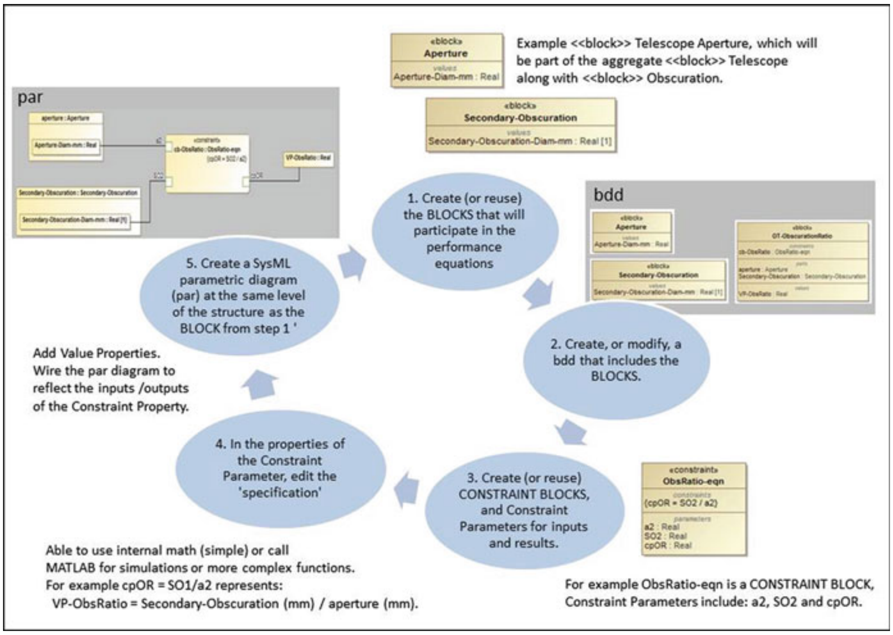

«block» Aperture
values
Aperture-Diam-mm : Real
Example <<block>> Telescope Aperture, which will be part of the aggregate <<block>> Telescope along with <<block>> Obscuration.
«block» Secondary-Obscuration
values
Secondary-Obscuration-Diam-mm : Real [1]
par
bdd
1. Create (or reuse) the BLOCKS that will participate in the performance equations
2. Create, or modify, a bdd that includes the BLOCKS.
3. Create (or reuse) CONSTRAINT BLOCKS, and Constraint Parameters for inputs and results.
4. In the properties of the Constraint Parameter, edit the 'specification'
5. Create a SysML parametric diagram (par) at the same level of the structure as the BLOCK from step 1 '
Add Value Properties.
Wire the par diagram to reflect the inputs /outputs of the Constraint Property.
«constraint» ObsRatio-eqn
constraints
{cpOR = SO2 / a2}
parameters
a2 : Real
SO2 : Real
cpOR : Real
Able to use internal math (simple) or call MATLAB for simulations or more complex functions.
For example cpOR = SO1/a2 represents:
VP-ObsRatio = Secondary-Obscuration (mm) / aperture (mm).
For example ObsRatio-eqn is a CONSTRAINT BLOCK, Constraint Parameters include: a2, SO2 and cpOR.

Fig. 25.7 Steps for building parametric relationships

- Create (or reuse) *CONSTRAINT BLOCKS*, and *Constraint Parameters* (see note above about not using hyphens if using MATLAB) for inputs and results. Reuse of *Constraint Parameters*, even for inputs seemed to cause unintended behavior.
- In the properties of the *Constraint Parameter* just created, edit the 'specification' using the mathematical format of MagicDraw, or the more complex call to MATLAB (see below), as required.

- Create a SysML parametric diagram (*par*) at the same level of the structure as the *BLOCK* from step 1, and including the *ValueProperties*, corresponding to the *BLOCKS* in the *bdd* from step 2. Wire the *par* diagram to reflect the inputs and outputs of the *Constraint Property*.

25.4.2 Performance Measures

To perform the calculations of the dependent attributes created by the procedure just described, it is necessary to create and initialize an instance of the *BLOCK* at the level desired. With multiple calculations, at a variety of levels of the structure, this can get a bit cumbersome. The performance attributes calculated by the model are:

- Span of GEO-belt accessible from a site (by longitude on GEO-belt)
- Total system coverage (all sites) of the GEO-belt ± 10 deg (see Fig. 25.8)
- Field of View of a optic/sensor combination (and whether limited by optic or CCD)
- Limiting magnitude for a given integration time with a given sky background (and whether limited by optic/sensor or site background) see Table 25.1.

The performance measures are calculated based on the specifications of the selected equipment, some user selected values, the properties of the targets, and properties of the location(s) of the observations.

MATLAB functions were developed to support all but the simplest calculations. As mentioned, several of the functions reused work from past research projects requiring only minor adaptation from scripts to functions and to output for use by ParaMagic. This effort showed how this earlier work could be integrated into the *BLOCKS*, constraints, and parametric description at the system level.

25.5 Conclusion/Discussion

The purpose of this prototype effort was to prove that it was possible to employ Model-based Engineering for the Teletrak system, while leveraging an extensive investment of analytical routines. These routines were previously written in MATLAB for complex analytical calculations and graphic products for a tradespace evaluation. The demonstration on the Teletrak case is encouraging for the next step, to scale up the model to perform a larger range of functions needed to improve the Space Surveillance Network (SSN). It is clear that the SysML model and parametric function structures can support an MBSE approach to unified

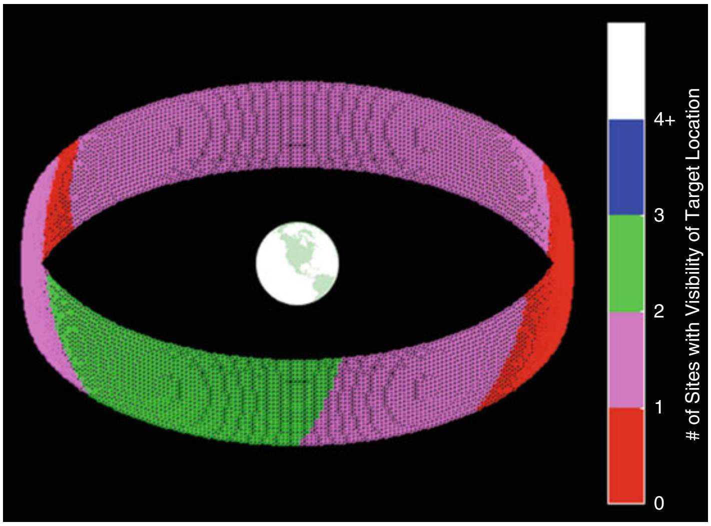


Fig. 25.8 MATLAB generated graphic showing the GEO-belt ±10 deg and the number of dedicated SSN observation sites with access

Table 25.1 Performance measures examples

Scope sensor	FOV deg	iFOV arcsec	min detectable magnitude
16in. ICX694	0.18 × 0.14 CCD	0.23	18.7
106 mm KAF16803	3.99 × 3.99 CCD	3.5	?
10in. 1CX694	0.28 × 0.23 CCD	0.37	17.8
106 mm ICX694	0.35 × 1.08 CCD	1.76	16.5

specification and performance verification for a wide variety of purposes. For Teletrak, the purpose was the specification of a new Wide Field of View (WFOV) system with a resolution commensurate with the seeing conditions in Dayton, Ohio. However, the power of the tools is illustrated by both their general utility to any optical and sensor pairing, and the independence of the calculation engine from the architecture modeling tool. While we were successful in discovering a route to build a model that would perform the functions desired, this effort also demonstrated some of the many limitations in, and between, the many software tools that compose the integrated functional model. Further, the issues encountered in the early stages of design and performance modeling are only the beginning of a full systems engineering lifecycle treatment, from preacquisition through sustainment and disposal, that will drive much greater demand for interoperable models and tools that can be transferred from one organization to another without losing fidelity or functionality.

Disclaimer The views expressed in this thesis are those of the author and do not reflect the official policy or position of the United States Air Force, Department of Defense, or the United States Government.

References

1. Estefan JA (2007) Survey of model-based systems engineering (MBSE) methodologies. INCOSE MBSE Focus Group 25(8)
2. Open Management Group (2015) System modeling language v1.4. Syst Model Lang. http://www.omg.org/spec/SysML/1.4/. Accessed 31 Oct 2016
3. NoMagic I (2016) Cameo/MagicDraw. Cameo/MagicDraw Prod page. http://www.nomagic.com/products/magicdraw.html. Accessed 20 June 2001
4. InterCax ParaMagic. ParaMagic Product page. http://intercax.com/products/paramagic/. Accessed 31Oct 2016
5. DoD/CIO (2010) The DoD architecture framework version 2.02
6. INCOSE (2016) Tools Integration and Interoperability Working Group. Tools Integr Interoperability Work Gr Home. http://www.incose.org/ChaptersGroups/WorkingGroups/technology/tools-integration-interoperability. Accessed 21 Jan 2016
7. Garrett RK, Anderson S, Baron NT, Moreland JD (2011) Managing the interstitials: a system of systems framework suited for the Ballistic Missile Defense System. Syst Eng 14(1):87–109. doi:10.1002/sys.20173
8. Schmunk MM. (2011) Initial determination of low Earth orbits using commercial telescopes
9. Gresham JL (2012) Optical closed-loop control tracking with commercial telescopes
10. Briggs GC (2012) Satellite detection and real-time orbit estimation with commercial telescopes
11. Moomey D (2015) Aiding geostationary space situational awareness using small aperture commercial telescopes
12. Gruss M. U.S. (2016) Air force: Russian rocket body may have broken up in orbit. SpaceNews, January 21
13. NASA (2015) Recent breakup of a DMSP satellite. Orbital Debris Q News 15(2)
14. National Academies Press (2012) Continuing Kepler's quest – assessing air force space Command's astrodynamics standards. National Academies Press, Washington, DC
15. NASA (2015) Satellite box score. Orbital Debris Q News 19(4)
16. Liou JC, Weaver JK (2005) Orbital dynamics of high area-to-mass ratio debris and their distribution in the geosynchronous region. In: European space agency, (Special Publication) ESA SP, pp 285–290. doi:10.1007/s13398-014-0173-7.2

Chapter 26
Model-Based Approach for Engineering Resilient System-of-Systems: Application to Autonomous Vehicle Networks

Azad M. Madni, Michael W. Sievers, James Humann, Edwin Ordoukhanian, Joseph D'Ambrosio, and Padma Sundaram

Abstract Autonomous Systems (e.g., self-driving vehicles) and autonomous systems network are becoming increasingly more feasible for real-world deployment with the advent of the Internet-of-Things (IoTs) and advances in sensing and machine reasoning technologies. Autonomous vehicle networks are system-of-systems and are amenable to model-based analysis and design. This paper presents a model-based approach for analyzing and designing resilient SoS subject to a variety of disruptions. The specific SoS addressed in this paper is an autonomous (i.e., self-driving) vehicle network. The approach employs deterministic and probabilistic modeling, and use-case patterns to model SoS behavior and explore mechanisms for introducing resilience into systems and SoS. Exemplar use cases for self-driving vehicles are provided to illustrate key aspects of the overall approach.

Keywords System design • Component-based design • Contract-based design • Interface

26.1 Introduction

"Autonomous" means having the ability for self-governance and independent operation. Autonomous systems (e.g., self-driving vehicles) are capable of exhibiting requisite performance with the desired level of reliability and safety under significant uncertainties in the real-world environment for extended durations, while compensating for system failures without external intervention.

A.M. Madni (✉) • M.W. Sievers • J. Humann • E. Ordoukhanian
University of Southern California, Los Angeles, CA, USA
e-mail: Azad.Madni@usc.edu

J. D'Ambrosio • P. Sundaram
General Motors, Detroit, MI, USA
e-mail: joseph.dambrosio@gm.com

Autonomous systems today are network-enabled and potentially capable of communicating with other vehicles and smart structures in the immediate vicinity (for collision avoidance) and distant vehicles and structures (for congestion management). Most autonomous vehicle concepts today employ a network connection to the cloud, other vehicles, and smart built-up structures. They do not make autonomous decisions with respect to destination and route selection.

Consumer acceptance of autonomous vehicles (AVs) depends largely on how they are introduced in the operational environment and the trust they engender. For example, AVs without a steering wheel and eventually without a human-in-the-loop are radical concepts that need to be ultimately acceptable to passengers. Fortunately, there is a precedent and lessons learned from the days when elevators were first introduced in buildings. In those days, elevators were operated by elevator operators who were responsible for ensuring that elevator cars stopped at passenger-selected floors. Elevator operators, responsible for opening and shutting the doors at different floors, occasionally made mistakes. However, these infrequent mistakes did not deter people in a hurry from leaping into elevator cars moments before the elevator operator shut the door. A missed step meant potential injury despite the presence of the elevator operator, who would not be able to react fast enough to stop the elevator.

It did not take long for the elevator industry to add doors with safety bumpers, and automate stopping. In 1900, elevators became "self-driving" in today's parlance. However, their introduction had a rocky start with uneasy passengers routinely stepping out of the elevator car to locate the "missing" elevator operator! The elevator industry, recognizing that it had a problem, concluded that people needed a fundamentally different experience to restore trust in elevators. With flurry of advertisements showing children and grandmas confidently riding elevator cars, they sought to dispel the mistrust in elevators. These interventions worked, and in the process produced valuable lessons. The first lesson was that eliminating human presence and human interaction/feedback does not imply eliminating all interaction. Having a reassuring voice piped into the car can go a long way to setting passenger expectations and building trust. The second lesson was that indiscriminate automation can produce unintended consequences and compromise system acceptance.

Today, with advances in sensing and the advent of IoT, and deep machine learning, AVs and AV networks have become possible. An AV network can be viewed as a system-of-systems (SoS) because it exhibits the characteristics of an SoS (Table 26.1) [1, 2].

An AV is capable of driving to its destination without human interaction, and with and without human occupants. Equipped with a sensor suite, preprogrammed logic, and deep machine learning algorithms, AVs are rapidly becoming capable of assessing and reacting to various traffic situations that involve other AVs and human-operated vehicles (including bicyclists and motorcyclists), and pedestrians. Today, AV designers such as Google are heeding lessons learned from the past to ensure positive passenger experience from the outset.

Table 26.1 An AV network is an SoS

Operational Independence of AVs
AVs operate independently as part of a traffic ecosystem
Managerial Independence of AVs
AVs governed independently while being part of traffic ecosystem
Evolutionary Development of SoS
Development and existence is evolutionary with functions and purposes added, removed, and modified with experience and need
Emergent SoS Behavior
AV-SoS performs functions and carries out purposes that do not reside in any single AV
AV-SoS behaviors are emergent – cannot be realized by a single AV
Geographic Distribution
AVs are displaced in space and time and primarily exchange information

26.2 Application of Model-Based Systems Engineering to AV-SoS

Model-Based Systems Engineering (MBSE) of AV-SoS begins with the development of an AV-SoS model. This model needs to be able to account for uncertainty, be verifiable, and be amenable to Test and Evaluation. In addition, the model needs to be able to handle nondeterminism, and possess sufficient flexibility to incorporate resilient responses to various types of disruptions. Finally, the model should scale with an increase in the size of network and heterogeneity of nodes. To ensure that these requirements are met, we begin by making certain simplifying assumptions to develop an understanding of the problem. The general problem cuts across multiple disciplines (e.g., SoS engineering, computer science, law, insurance, ethics, and public interaction) [3]. We confine our discussion in this paper to the technical problem (i.e., modeling, analysis, and design of a resilient AV-SoS).

The specific simplifications we introduce in AV-SoS modeling are: (a) dedicated lanes for AVs (analogous to high-occupancy vehicle lanes); (b) all vehicles in the initial set of use cases are AVs; (c) the level of autonomy of the AVs is SAE level 4; (d) safe and resilient behaviors are confined to a defined set of disruptions; (e) the operational environment is defined using a core set of environmental variables that are used to define the perturbations; and (f) SoS architecture will allow for human-driven vehicles to be introduced late into the SoS network for more comprehensive modeling, analysis, and design activities. The introduction of human-driven vehicles also introduces human errors, a rich source of additional disruptions. With these simplifications, MBSE can be discussed for AV-SoS networks. MBSE is the formalized application of modeling to support system requirements definition, system design, system analysis, verification and validation, and test and evaluation. It begins with the conceptual design phase and continues through development and later life cycle phases [4].

In the following illustrative AV-SoS example presented, the goal of MBSE is to enable the modeling and analysis of the AV-SoS network and introduction and

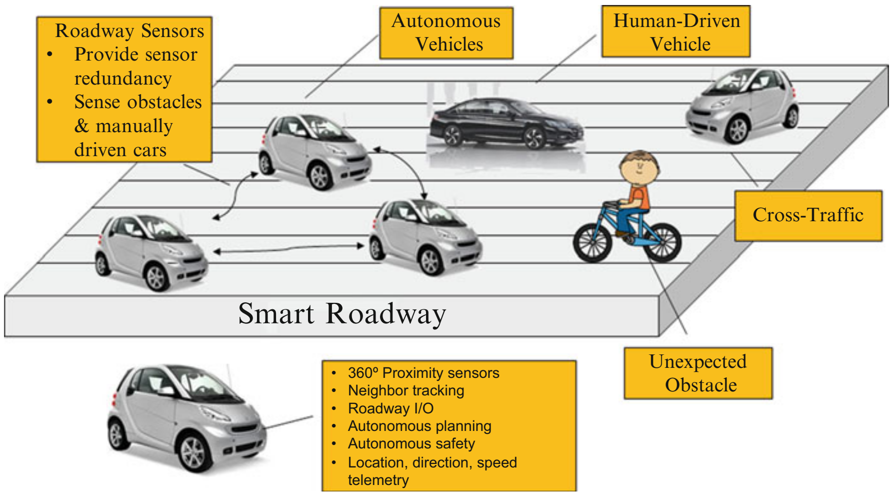


Fig. 26.1 Problem context for resilient AV-SoS network design

evaluation of resilience techniques for various types of disruptions. In this example, networked AVs, equipped with sensors, exploit sensing and networking capabilities to replan, reorganize, and adapt to internal and external disruptions. The AVs are capable of interacting with each other and with other entities (e.g., smart static structures, human-driven cars). The AV-SoS and each AV employ information acquired from the SoS network in local and global resilience algorithms to avoid, survive, and learn from disruptions. A representative problem context for exploring and assessing AV-SoS resilience is presented in Fig. 26.1 [5].

In this problem context, self-driving vehicles are required to safely negotiate routine hazards, unexpected obstacles, cross-traffic, and human-driven cars. The resilient behavior needed in these vehicles includes the ability to respond to internal and external disruptions at both individual and the AV and the AV-SoS levels. The overall concept of operations (CONOPS) is as follows. An AV receives inputs from a variety of sensors on stationary objects and other vehicles. This information is used to adapt behavior (i.e., exhibit resilience) at both the individual vehicle and the SoS levels (Fig. 26.2).

26.3 Real-World Use-Case Example

The Google car rear-ender early in 2016 offered an excellent example of a real-world use case. The Google AV was rear-ended by a bus when it attempted to change lanes to avoid sandbags without accounting for the existence of the bus. In a fully autonomous SoS network, the vehicle would have been able to coordinate

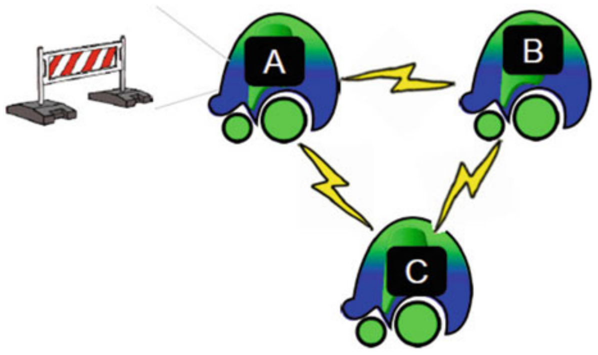


Fig. 26.2 Real-world example: Google car rear-ender

actions and avoid the collision. This example provides the requisite information to define an exemplar use case, along with alternative use cases.

Exemplar Use Case: All vehicles in the SoS network (comprising vehicles A, B, and C) are autonomous vehicles (i.e., SAE level 4 or 5). Vehicle A senses an obstacle, and wants to move into the lane occupied by Vehicle C. Vehicle A communicates its intent to Vehicles B and C, to verify safety of proposed lane change. If Vehicles B and C confirm that lane change is safe, Vehicle A performs the lane change. If lane change is unsafe, then Vehicle A applies brakes.

Alternate Use Case #1: Vehicle C is not autonomous. In this case, Vehicle A signals Vehicle B, and proceeds to apply brakes.

Alternate Use Case #2: Neither Vehicle B nor Vehicle C is autonomous. Vehicle A updates its speed based on proximity of Vehicles B and C. Then Vehicle A may only cross into Vehicle C's lane, if sufficient, reliable information about Vehicle C's location and speed is available.

To summarize, the foregoing has presented a basic use case, and two variants of the basic use case. The basic use case is based on a fully autonomous SoS network in which all vehicles are capable of communicating with each other, and jointly determining how best to respond to unusual events or disruptions. The variants of this basic use case employ one or more human-driven/nonautonomous vehicles. In these cases, the AV must account for human reaction times as well as location, speed, and reliability of information about the nonautonomous vehicles in decision making.

The goals of the AV-SoS are to:

- Satisfy transportation needs of human occupants
- Minimize occurrence of accidents
- Minimize communication and communication dependencies
- Minimize traffic bottlenecks (i.e., maximize flow efficiency)
- Maintain safe following distances to prevent accidents and perform safe maneuvers

The introduction of resilience in the SoS network begins with a set of use cases. Once again, for simplicity, let us assume a fully autonomous SoS, that is, all vehicles in the SoS are AVs. Let us also assume that SoS planning is hierarchical, and the communication protocols followed by the AVs are:

- *Preplanned:* vehicles follow preplanned rules
- *Context-driven:* a limited set of AVs are correlated based on a use-case pattern

Preplanned behaviors of AVs are analogous to mission system tests (MST) performed for spacecraft missions, while context-driven behaviors of AVs are similar to operational readiness tests (ORTs) that are also performed for spacecraft missions. MSTs are performed when we know the answer, and are interested in verifying that the SoS does what is expected. ORTs are performed when we do not know the answer but are interested in verifying that the SoS remains safe.

When it comes to SoS resilience, it is important to realize that AVs will employ different parameters and/or rules in resilience algorithms than those used in automated, human-driven vehicles. As important, the SoS will be capable of determining which use case applies, and accordingly make changes to model parameters.

26.4 AV Behavior Patterns Within SoS Network

Defining use cases for AV-SoS is challenging because there are numerous scenarios to choose from, with several that do not correlate with today's driving scenarios. Therefore, the choice of scenarios should be based on how well they support innovation in generating resilient responses, and how well they enable AV-SoS testing. Complicating factors in the scenarios are associated with the heterogeneous environment. These include: legal (e.g., right turn on red laws, jaywalking, crosswalks, time windows (e.g., HOV lanes, parking); informational (e.g., presence of signs, color and type of signs or signals, marking of emergency vehicles crosswalk marking); and maintenance of infrastructure and neighboring systems (e.g., faded road lines, graffiti in signs; broken lights, poorly maintained human-driven cars on the road). And finally, choosing the autonomy level (from SAE-defined autonomy levels) impacts SoS behavior complexity. By choosing SAE level 4, we can avoid the complex issues of human–system handoffs in vehicle control. Against the foregoing, we can model nominal and off-nominal (predictable) AV operations (Figs. 26.3 and 26.4). Figure 26.3 shows the closed loop process associated with nominal (and predictable) autonomous vehicle operation. Beginning with user goal and preferences, the system evaluates route options and road conditions, chooses the "best" route, contacts vehicles in the vicinity, executes plan when safe, continues on with plan and telemetry I/O while avoiding collision, modifying operating parameters, and stopping upon arrival at destination. Figure 26.4 provides the plan segment associated with off-nominal (but predictable) AV operation. The control architecture of an AV is shown in Fig. 26.5.

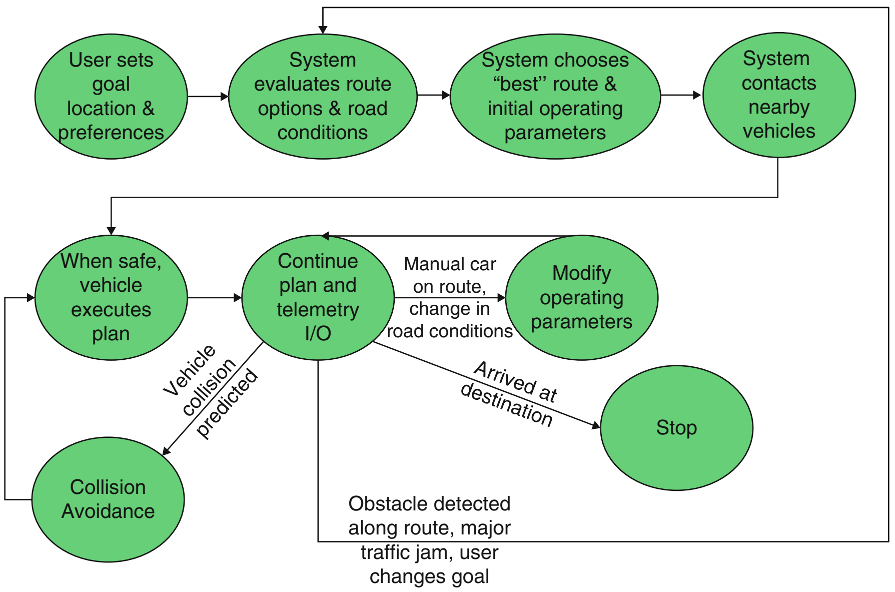
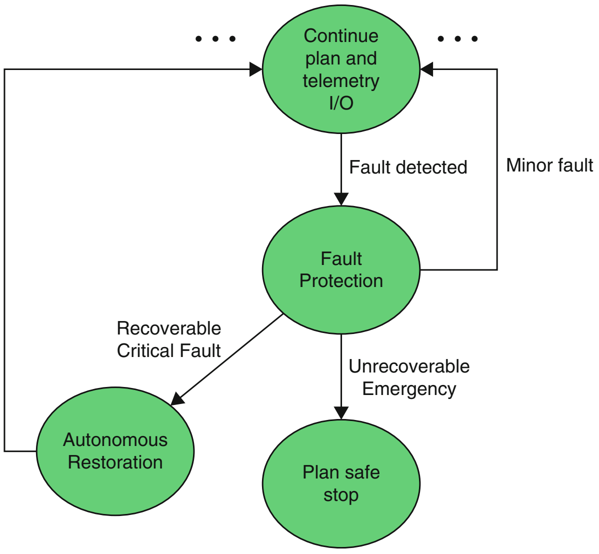

User sets goal location & preferences
System evaluates route options & road conditions
System chooses "best" route & initial operating parameters
System contacts nearby vehicles
When safe, vehicle executes plan
Continue plan and telemetry I/O
Manual car on route, change in road conditions
Modify operating parameters
Vehicle collision predicted
Arrived at destination
Stop
Collision Avoidance
Obstacle detected along route, major traffic jam, user changes goal

Fig. 26.3 Nominal (predictable) autonomous vehicle operation

Fig. 26.4 Off-nominal (predictable) autonomous vehicle operation



The vehicle control plan comprises location-dependent commands that are sent to the vehicle controller. The vehicle controller accepts state estimates and updates configuration goals that are sent to the deductive controller. The deductive controller accepts configuration goals (from the vehicle controller), inputs from the vehicle model, observations (from the vehicles), and environmental inputs (from the

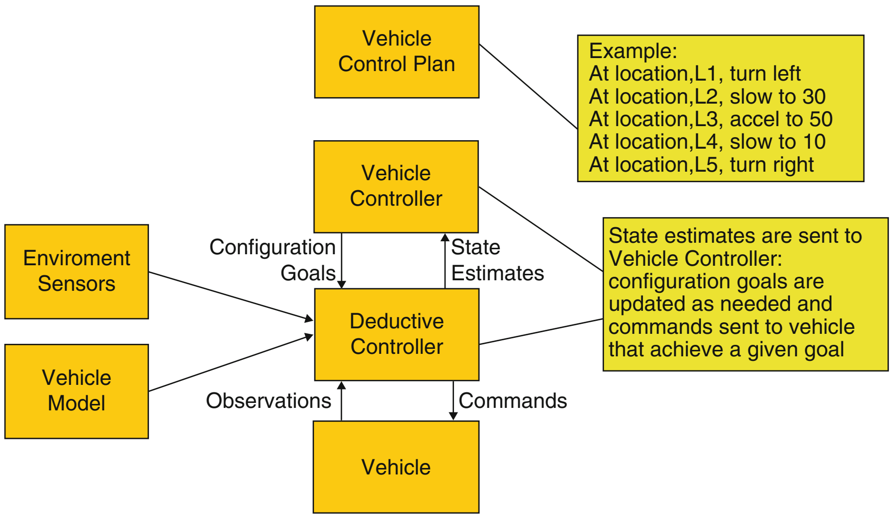

Vehicle Control Plan
Example:
At location,L1, turn left
At location,L2, slow to 30
At location,L3, accel to 50
At location,L4, slow to 10
At location,L5, turn right
Vehicle Controller
State estimates are sent to Vehicle Controller: configuration goals are updated as needed and commands sent to vehicle that achieve a given goal
Enviroment Sensors
Configuration Goals
State Estimates
Deductive Controller
Vehicle Model
Observations
Commands
Vehicle

Fig. 26.5 Autonomous vehicle control architecture

environment sensors), to produce state estimates to the vehicle controller, and vehicle commands that are sent to the vehicle to achieve a given goal.

Use cases for AVs are associated with two types of patterns: those that are experienced by human-driven cars; and those that apply exclusively to AVs. Table 26.2 presents an overview of AV behavior patterns that can be used to develop AV use cases.

The following paragraphs describe the different AV behavior patterns.

1. Highway Merge (on-ramp or off-ramp)

Merging is a coming together or blending of vehicles to maintain a smooth flow of traffic. The main challenge in performing this behavior is sensory in that the merging vehicle cannot always see oncoming traffic. In human-driven cars, humans “judge” relative speeds to merge. Introducing this capability in self-driving cars is a challenge? A further complication is that merge difficulty varies as a function of traffic and road/type quality. Merge behavior tests fore, aft, and lateral car control and sensing coupled with dynamic uncertainty.

2. Four-Way Stop

A four-way stop is associated with a four-way intersection of two-way roads controlled by stop signs or road markings. This context is different from stoplights where lights protect certain maneuvers. A four-way stop requires negotiation between self and other vehicles in accord with “right-of-way” rules. The problem is relatively simple when all vehicles are autonomous. However, when the vehicles in question are a mix of AVs and human-driven cars, the problem becomes challenging. The four-way stop tests dynamic decision making in uncertain, heterogeneous (i.e., AV and human-driven vehicle) vehicle environments. Human-

Table 26.2 Patterns-driven AV use cases

Highway merge (on-ramp or off-ramp)
Four-way stop, heterogeneous systems
Lane blockage
Unprotected left turn
Highway driving
Rural/country road driving
City driving (environmental density)
Scenario modifiers and injects
Weather Yielding for emergency vehicles Parking (garage, roadside, lot) Pulling over for: police/mechanical/weather issue

drivers rarely follow the "letter of the law" in these cases, but still manage to negotiate such challenging situations safely.

3. Lane Blockage

This is a common occurrence on a two-way road. Examples of static blockage are: disabled vehicle, felled tree, construction, and weather effects (e.g., flash flooding). Examples of dynamic blockages/obstacles include traffic accident, pedestrian, animal, or child that darts into the lane. Successful execution of an avoidance maneuver requires environmental reasoning with respect to rules of the road and new developments that require an avoidance or stopping maneuver; and adaptive decision making such as the AV deciding to cross the yellow line for a brief moment using sensory data to perform a maneuver outside the norm.

4. Highway Cruise

This pattern is associated with multilane highway driving. It primarily tests navigational and vehicular controls. Even in dense environments, objective variance among vehicles (agents) is quite small due to road-type restrictions. This maneuver is relatively straightforward in comparison to others. This pattern can serve as the baseline scenario for introducing "injects" such as hazards, weather changes, merging, scaling traffic, and responding to vehicle emergency.

5. Unprotected Left Turn

This pattern is associated with a T-intersection in which the perpendicular road does not have a stoplight to ensure protected left turns. The challenge tests that the AV needs to meet are judging openings based on multisensory data, assessing risk, and understanding and predicting the behaviors of other vehicles. This pattern can be used for scenario injects (e.g., accident avoidance, how to handle a looming accident).

6. Rural/Country Road Cruise

This pattern, which is associated with a two-way road, is generally more difficult and riskier than highway cruise. This is because physical lines-of-sight vary as do road quality and type. The tests associated with this pattern that an AV needs to pass are environmental awareness at speed, sensing in diverse and dense environments, and communication capabilities. This pattern serves as a baseline to inject other events similar to those for the highway cruise pattern (use case).

7. City Driving

This pattern is associated with driving in dense areas comprising automobiles, infrastructures, pedestrians, motorcyclists/cyclists, traffic police officers, construction, trains, buses, and roadside parking. This pattern is also associated with dynamic environments in which other than infrastructure, most elements move and involve humans. The third characteristic is fluctuating speeds, that is, transition from slow to fast and back to slow. The challenges confronting this behavioral pattern are data handling and sensor capabilities in data-rich environments, partitioning data by importance, and making local decisions in light of holistic objective and full consideration of available data.

26.5 Use-Case Modifiers (“Injects”)

Several factors can serve as modifiers in use cases. One of the most important modifiers is “weather.” Inclement weather affects several variables used to define use cases. These include: sensor capability, vehicle control capability, route planning (e.g., avoidance, detour), environmental awareness, decreased performance of neighboring vehicles, and AV decision making (navigational, abort mission criteria, uncertainty). A study by Wachenfeld et al. (2016) defines success for driving in weather as “The quality as well as the success rate with which the driving robot performs the driving task is similar to the human quality and success rate” [6]. For example, ceasing operation due to weather occurs only “when a driver would discontinue the journey as well” [6].

Other “injects” into use cases include: yielding to emergency vehicles, parking, and pulling over. Yielding to emergency vehicles tests sensor capabilities and communications capabilities (long haul and across the ad hoc mesh) as well as environmental reasoning while counterintuitively maneuvering the vehicle (e.g., pulling off road on unpaved shoulder to let emergency vehicle pass). Parking can be its own use case, or added as a preuse case or postuse case (i.e., additional activity) to an existing use case (pattern). This use case tests environmental deduction and induction about the state of vehicles – which vehicles are parked, which are standing, and which are waiting for a parking spot. The pulling over use case is associated with a mechanical malfunction or fault, or other emergency/unusual situation.

26.6 Formal Modeling of AV-SoS

Formal methods have been in use in the chip design industry for decades. Formal modeling approaches introduce structure and rigor in model representation, and facilitate verification, test, and evaluation. In essence, formal methods are especially suited for system and SoS development. Today, formal methods are being used to model cyber–physical–social systems and system-of-systems. Their key distinguishing characteristic is that they exploit notations with mathematically rigorous semantics. This property ensures that analyses performed using such models offer a high degree of confidence that the models are correct. Model correctness implies model completeness, consistency, and traceability. The set of questions that the model is expected to answer defines the intended purpose of the model.

With respect to system/SoS modeling, formal methods enable the systems engineer to specify the system/SoS rigorously, and analyze the resultant model to ensure that the specifications are consistent and reasonably complete. When this is the case, it can be guaranteed that the system/SoS model possesses the requisite properties. This is basically the "assert-guarantee" construct that underpins several formal modeling approaches such as contract-based design. As important, the application of formal methods tends to reduce the risks in the development process by reducing the likelihood of late detection of defects, as well as reducing the cost of defect detection.

Static analysis approaches enabled by formal modeling include model checking and formal theorem proving. These methods when applied to models created in upfront engineering help identify areas of incompleteness and ambiguity in requirements and/or specifications. When applied to later stages in design or code-level analysis, they can be used to identify specific patterns of defect. Formal methods enable system developers to prove mathematically that the system/SoS models that they have developed have specific essential properties (e.g., resilience, security).

Formal methods are important for AV-SoS model verification, test, and evaluation. Since an SoS is a software-intensive system (SIS), it is not possible to demonstrate that it is error-free by testing it. This is because testing every possible path through the software code, for every combination of data that could potentially cause that path to fail, and then checking that each path led to a correct result, is impractical. It would take an infinitely long time and incur prohibitive costs even if it were possible. This is in sharp contrast to physical systems (e.g., aircraft wing) that obey laws of physics that allow engineers to test a design at extreme values, and infer behavior between the extremes [7]. In a software-intensive SoS, the behavior of the SoS depends largely on software logic and not physical laws (note: the behavior of an individual AV does depend on physical laws). Thus, for an SoS, it cannot be assumed that because the software logic functions as intended for some input values, it will function as intended for other input values, even if the latter lie between the extremes defined by the former.

The implication of the above is that SIS's invariably contain latent errors, even after rigorous testing during development. Research studies in academia and industry have shown that there are typically 5–30 errors in 1000 lines of code at the time that the SIS goes into service, following normal testing. This is where the value proposition of formal methods comes into play.

With the use of formal methods even for a fraction of the development process, it becomes possible for software tools to analyze models and detect errors more quickly, inexpensively, and comprehensively than with conventional testing methods. Formal methods are capable of supporting automated testing (i.e., test generation, test execution, test output checking). This, in essence, is "model-based testing" with the added rigor provided by formal semantics. In the software world, formal model-based testing has shown significant time and cost savings when compared to traditional manual testing.

Modeling an AV and an AV-SoS is the starting point for the design of the AV-SoS. There are several requirements that the selected modeling constructs need to satisfy: enable verification of correctness (i.e., completeness, consistency, and traceability); ensure scalability with the number of AVs in the network; support test and evaluation; and ensure satisfaction of desired design quality attributes.

For an SoS, several formal methods can be employed that range from deterministic to probabilistic. Examples of deterministic methods are computational tree logic (CTL); linear temporal logic (LTL); and Contract-Based Design (CBD). Examples of probabilistic or stochastic methods are Hidden Markov Models (HMM), and Partially Observable Markov Decision Processes (POMDPs). Deterministic models are useful in verification and testing, while probabilistic models are helpful in dealing with partial observability of the SoS states and uncertainty in the environments and SoS parameters.

With the above requirements in mind, we employ a modeling approach consistent with a Sense–Plan–Act (SPA) process that combines deterministic and probabilistic approaches. Specifically, we combine CBD and POMDP. CBD is a formal method for explicitly defining, verifying, and validating system requirements, constraints, and interfaces. With CBD, an implementation satisfies a design constraint if it fulfills guarantees when assertions are true. In the CBD approach, design is a composition process, in which the system design is the result of successive assembly of components. A component contract specifies assumptions about its environment, and guarantees about its behavior. A composition is said to be "well-formed" if the assumptions of each component are contained in guarantees offered by others. The appeal of CBD stems from the fact that statements in a contract are mathematically provable. The limitation of CBD is that the assertions are invariant. From a resilience perspective, we must relax the requirement for invariant assertions and introduce flexibility by way of POMDP models. A POMDP implements SPA by continuously taking actions based on maximizing a reward function over a set of belief states and then updating the belief state after taking an action.

The flexible CBD paradigm specifies the dynamic behavior of an SoS by a set of atomic events in which actions taken alter the SoS state. Due to the large number of states, managing a flat SoS state space model is not possible. We instead create a

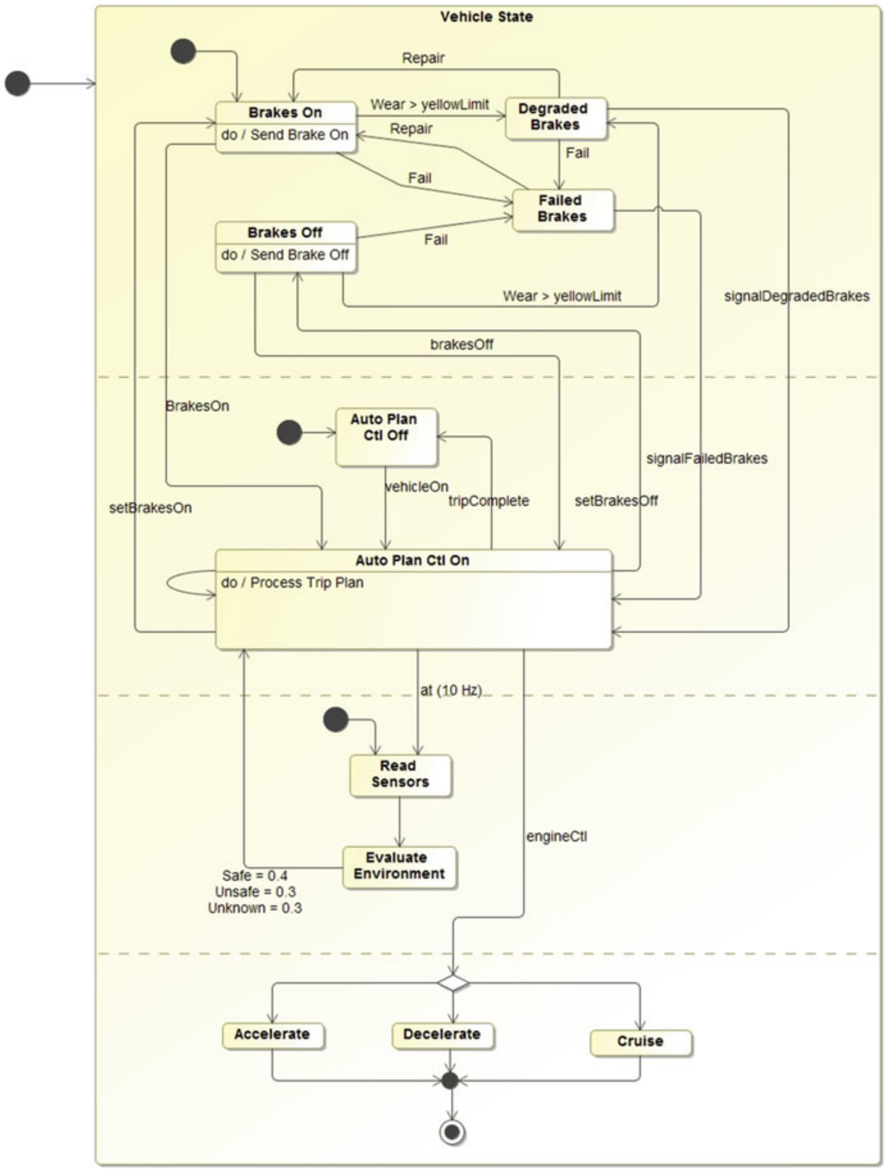


Fig. 26.6 Example flexible CBD state representation for an autonomous automobile in SysML

hierarchical construction in which guarantees become the input assumptions for peer- and higher-level guarantees as shown by a conceptual autonomous vehicle system in Fig. 26.6. Figure 26.6 depicts invariant contracts in which assertions are either "true" or "false" and trigger invariant actions. For example, the assumption "brake wear > 'yellow' limit" triggers "setDegradedBrakes." There are also contracts related to safe, unsafe, and unknown environment conditions that are probabilistic and may invoke any one of a number of actions depending on the seriousness of the situation and the confidence in knowing SoS state. For example, we may have estimated that the probabilities of "safe," "unsafe," and "unknown" are 0.4, 0.3, and 0.3, respectively. We might also have two actions: take more samples, and "emergency stop." Based on traffic conditions, weather, vehicle speed, etc., a reward function might be maximized by the action, "take more samples." After more samples are collected, the state probabilities might change in favor of "unsafe," and the best option then becomes, "emergency stop."

The Autonomous Planning function in Fig. 26.6 satisfies a number of short, medium, and long-term contracts consistent with a time-to-critical-effect (TTCE). TTCE is the time between the onset of a disruption and the point at which the disruption becomes a serious hazard if not managed. SPA cycle time must be shorter than the TTCE for each disruption; however, those times will be functions of vehicle operation, environmental conditions, and the type of disruption. Consider, for example, a stopped vehicle contrasted with another traveling at highway speed on a wet road. A Planning function faced with a degraded brake indication while stopped can disable vehicle acceleration and take as much time as necessary for resolving the brake issue. However, the other vehicle will need high-rate knowledge of brake and road conditions so that it can safely reduce vehicle speed and exit traffic lanes. The implication illustrated by this example is that the reward functions and actions change with the belief state determined by POMDP contracts within the Planning function.

Contract-based design (CBD) is a domain-agnostic, methodology-neutral approach for modeling complex systems and managing complexity. Its most popular use is in requirements engineering. CBD has also been successfully employed in virtual integration and deployment. CBD provides rigorous scaffolding for verification, analysis, and abstraction/requirement. Since contracts are applied in the early stages of system design, contracts need to support domains in which automated reasoning does not apply (e.g., due to issues of decidability). CBD does not need to be fully automatic. The combination of manual local reasoning with automatic lifting of local to system-wide properties is "sensible" and well-supported by contracts [8]. Contracts are also ideal to solidify vertical processes (i.e., abstraction/refinement) and horizontal processes (i.e., composition/decomposition) providing the theoretical bases for supporting formal methods.

The rationale for choosing POMDP stems from the fact that even though many real-world problems and systems are complex, and only partially observable and controllable, the Markov assumption is usually valid. A POMDP model comprises a set of states S, a set of actions A, and a set of observations O. A POMDP comprises a transition model, a reward model, and an observation model. The Markov assumption applies to the transition model, with the optimal policy depending only on the current state. POMDP is an effective modeling approach when the operational environment is only partial observable, with the current state usually not known. Therefore, the agent (i.e., autonomous system) cannot execute the optimal policy for that state. In the proposed approach, contracts become flexible through relaxation of time-invariance restriction on the state space and action space, adding an evaluation metric to POMDP to determine the best action updating emission and transition probabilities of the hidden state, and finally adding the concept of time to POMDP. Then the CBD without time-invariance restriction is augmented with POMDP. We call the resultant hybrid modeling contract, which extends deterministic contract to represent stochastic systems, a resilience contract (RC) [9–11]. The key features of the RC are in-use learning, uncertainty handling, and pattern recognition. A resilience contract, which is specified during system design, is subsequently "trained" when the system is put to use ("learning").

26.7 Outlook for the Future

This paper has presented a model-based approach for engineering resilient SoS. The target domain is networked autonomous (i.e., self-driving) vehicles. Networked autonomous vehicles are a system-of-systems (SoS) and, thus amenable to being addressed by model-based techniques for modeling, analyzing, and designing SoS. The paper presents a new modeling construct, called the resilience contract (RC). An RC is a combination of contract-based design with Partially Observable Markov Decision Processes (POMDP). The CBD component, based on "assert-guarantee" paradigm, is well-suited for solution scalability and supporting test and evaluation. The POMDP component, the probabilistic part of RC, is well-suited to introducing flexibility in CBD and enabling contracts for dynamic environments in which the agents need to be controlled in a partially observable, uncertain environment. Specific use cases are presented to illustrate how the overall approach can be applied to networked autonomous vehicles. The overall approach is applicable to any multiagent problem requiring multiagent control in an uncertain, partially observable environment. Advances in MBSE approach for SoS will pave the way for modeling, analyzing, and designing SoS with specific quality attributes. In particular, the concept of an RC will provide the right balance in satisfying verification and validation and test and evaluation needs, while also assuring the model has the requisite flexibility to develop resilient systems and SoS.

Acknowledgment The authors acknowledge the inputs of CPT Kurt Klingensmith who provided exemplar use cases.

References

1. Madni AM, Sievers M (2013) System of systems integration: key considerations and challenges. Syst Eng 17(3):330–347
2. Madni AM, Sievers M (2013) Systems integration: key perspectives, experiences, and challenges. Syst Eng 17(1):37–51

3. Madni AM (2007) Transdisciplinarity: reaching beyond disciplines to find connections. J Integr Des Process Sci 11(1):1–11
4. INCOSE (2007) Systems engineering vision 2020 (INCOSE-TP-2004-004-02). http://www.incose.org/ProductsPubs/pdf/SEVision2020_20071003_v2_03.pdf
5. Madni AM (2017) Transdisciplinary systems engineering: expoliting convergence in a hyper-connected world. Springer, New York
6. Wachenfeld W, Winner H, Gerdes JC, Lenz B, Maurer M, Beiker S, et al (2016) Use cases for autonomous driving. In: Autonomous driving. Springer, Berlin/Heidelberg, pp 9–37
7. The Institute of Engineering and Technology (2011) Formal methods: a factfile
8. Bienveniste A, Caillaud B, Nickovic D, Passerone R, Raelet J, Reinkemeier P, Sangiovanni-Vincentelli A, Damm W, Henzinger T, Larson K (2015) Contracts for systems design: methodology and application cases. Inria research report no. 8760
9. Madni AM, Sievers M (2016) Model based systems engineering: motivation, current status and needed advances systems engineering
10. Madni AM, Sievers M (2015) A flexible contract-based design framework for exaluating system resilience approaches and mechanisms. IIE annual conference and expo, Nashville, May 30–June 2
11. Sievers M, Madni AM (2014) A flexible contracts approach to system resiliency. Systems, Man and Cybernetics (SMC), 2014 I.E. international conference, IEEE

Chapter 27
Validation and Verification of MBSE-Compliant CubeSat Reference Model

David Kaslow and Azad M. Madni

Abstract As Model-Based Systems Engineering (MBSE) continues to mature and becomes part of space engineering practice, the concept of a Reference Model becomes increasingly important. The CubeSat Reference Model (CRM) is an example of a reference model that is being developed by the INCOSE Space Systems Working Group (SSWG). The intent of the model is to facilitate the design, verification, and validation of CubeSat design. The CRM is being developed with sufficient flexibility to support customization for specific CubeSat missions by mission-specific CubeSat teams. This paper presents the key elements of the CRM developed using MBSE practices. It presents different views of the model along with a validation and verification approach. Further research is needed into how best to augment with other models to facilitate CubeSat test and evaluation.

Keywords Verification • Validation • CubeSat • Reference model • MBSE

27.1 Introduction

A CubeSat is a low-cost standardized nanosatellite. It originated from the CubeSat Project that was established in 1999 by California Polytechnic State University (Cal Poly), San Luis Obispo and Stanford University's Space and Systems Development Laboratory (SSDL). The CubeSat Project was established for the university community to design, build, and launch satellites using mostly off-the-shelf components. The basic CubeSat unit is 10 × 10 × 10 cm with a mass of about 1.3 kg. This cubic unit is referred to as 1U. CubeSat units can be joined to form a larger satellite. One-, two-, and three-unit (1U, 2U, and 3U) CubeSats have been the most common configurations so far. They are typically launched as secondary payloads or deployed from the International Space Station.

D. Kaslow (✉)
INCOSE Space Systems Working Group Chair, San Diego, CA, USA
e-mail: david.kaslow@gmail.com

A.M. Madni
University of Southern California, Los Angeles, CA, USA
e-mail: Azad.Madni@usc.edu

A.M. Madni et al. (eds.), *Disciplinary Convergence in Systems Engineering Research*, https://doi.org/10.1007/978-3-319-62217-0_27

Model-Based Systems Engineering (MBSE) is a key recent practice to advance the systems engineering discipline [1]. The International Council on Systems Engineering (INCOSE) established the MBSE Initiative [2] to promote, advance, and institutionalize the practice of MBSE. As part of this effort, since 2011 the INCOSE Space Systems Working Group (SSWG) Challenge Team has been investigating the applicability of MBSE for designing CubeSats. The SSWG team comprises academics (including faculty and students), practitioners (including engineers and software developers from NASA centers and industry), and representatives of commercial tool vendors.

The goals of the MBSE Challenge Project are to:

- Demonstrate Model-Based Systems Engineering (MBSE) methodology as applied to a CubeSat mission
- Provide a CubeSat Reference Model (CRM) for CubeSat teams to use as the starting point for developing mission-specific CubeSat models
- Develop the CRM as an Object Management Group (OMG) specification

Central to this MBSE work is the creation of a CubeSat Reference Model (CRM). A generic reference model in MBSE is a conceptual framework based on a domain-specific ontology. It consists of an interlinked set of concepts produced by a body of subject matter experts with the express purpose of fostering clear communications within collaborative teams of stakeholders. A reference model embodies the set of core concepts (including goals) that engineers can use for various purposes such as verification and validation, and test and evaluation. An apt analogy, according to Madni [3, 4], is that of examining a precious stone such as a diamond. “As you hold the gem up to the light and rotate your hand, you get to view its different facets. The rays of light from each facet offer a unique insight about the diamonds clarity and cut” [3]. A reference model is much like a gem. Its facets correspond to different perspectives or views, while the rays of light correspond to the information and insights contained in each perspective view. A reference model is a collection of different perspectives. It offers a high-level view of the problem domain (space). The key characteristics of a reference model are: abstractions, relationships, and entities. A reference model is methodology-neutral and technology-agnostic [3, 4].

Thus, the CubeSat Reference Model is an abstract framework for understanding the relationships among the entities of the CubeSat environment. Even though it is at least three levels of abstraction away from any physical instantiation, it still provides valuable assistance to the mission-specific CubeSat development teams.

Our use of the term Reference Model should not be confused with the use of the same term by the Organization for the Advancement of Structured Information Standards (OASIS). OASIS uses the term in the context of Service-Oriented Architectures [5].

The term “Model” as used in Model-Based Systems Engineering (MBSE) is represented using a language such as Systems Modeling Language (SysML). Since we are interested in modeling a generic CubeSat mission domain, and not a specific mission, we qualify the model as a “Reference Model.”

The CRM will be delivered by SSWG to mission-specific CubeSat development teams to:

- Populate the model with their Mission Statement and Mission Objectives, their Measures of Effectiveness and Performance, and their Stakeholders and Stakeholder Concerns
- Develop their Use Cases, System and Subsystem requirements, and Validation and Verification approach
- Refine the Logical Architecture and develop the Physical Architecture

The challenges [6] facing the development teams include the following:

- There is currently no agreed-upon process for developing a CubeSat Enterprise [7].
- To date, no mission-specific CubeSat Enterprise has been modeled.
- It is anticipated that different mission-specific CubeSat development teams will bring different levels of engineering and development skill and experience to bear.
- Budget constraints will continue to be a driver.

Encouraging the various mission-specific CubeSat development teams to adopt MBSE, Object-Oriented System Engineering Methodology (OOSEM), and SysML and providing them a standard Reference Model for their development will boost the efficiency and productivity of less experienced teams without any negative impacts.

Previously, the SSWG demonstrated the ability to model behaviors, interface with commercial off-the-shelf (COTS) simulation tools, and carry out trade studies [8]. Currently, the team is building a reference CubeSat model for use by aerospace students in the classroom, and by mission teams building mission-specific CubeSats [9–14].

In Sect. 27.1, we have introduced the SSWG CubeSat effort. Section 27.2 presents a more detailed look at the effort required to develop the CubeSat Reference Model, including a description of the Logical Architecture, and the organization of the model (particularly the requirements captured in the model). Section 27.3 identifies the effort required to develop a Mission-specific CubeSat model using the CRM as a foundation. Section 27.4 discusses the approach to conducting Validation and Verification of the CubeSat Reference Model, and compares that approach to the approach for conducting Verification and Validation of the Mission-specific CubeSat model. Section 27.5 presents conclusions and future work.

27.2 CubeSat Reference Model Development [13]

The CRM is intended to be used by university project teams designing space missions utilizing the CubeSat form-factor. The model is being developed assuming that the members of the team have an intermediate-level understanding of space mission analysis and design, Model-Based Systems Engineering (MBSE), Systems Modeling Language (SysML), and that their work is being guided by subject matter experts. MBSE is the formalized application of modeling to support key systems engineering tasks for addressing requirements, design, analysis, and validation and verification.

Logical and Physical Architectures. The CRM provides the logical architecture of a CubeSat. The logical components, which are abstractions of physical components, symbolically execute system functions (i.e., without implementation constraints). The physical architecture defines physical components of the system including hardware, software, persistent data, and operational procedures. The CRM logical elements are intended to be reused as a starting point for a mission-specific CubeSat logical architecture, followed by the development of physical architecture from the logical architecture during CubeSat development. On the other hand, should the mission-specific team decide to adopt a different logical architecture, the CRM is sufficiently flexible to accommodate this change.

CubeSat Domain and Enterprise Figure 27.1 shows the CubeSat Domain, which consists of the CubeSat Mission Enterprise (with its two segments and various services), Stakeholders, External Environment, and External Constraints. The External Environment consists of the Space Environment and Earth Environment. The External Constraints include Licenses and Regulations. The CubeSat Mission Enterprise encompasses everything that involves the development, deployment, and operation of the CubeSat mission.

CubeSat Space Segment The CubeSat Space Segment consists of one or more CubeSats along with their orbits and subsystems. The Space Segment includes designs, interfaces, and operations to comply with the requirements and constraints that are imposed by the External Environment, as well as those by other aspects of the mission such as the Transport, Launch, and Deploy Services. For example, a launch has a pressure and vibration profile that constrains the design of the CubeSat. These requirements and constraints can be incorporated into a Transport, Launch, and Deploy Services model unique to the service providers.

CubeSat Ground Segment The CubeSat Ground Segment consists of the CubeSat Mission Operations and one or more Ground Stations. Mission Operations includes the software, data, procedures, and personnel used to operate the CubeSat mission. Mission Operations activities include mission planning and scheduling, command and control of the CubeSat, control of the ground equipment, mission telemetry processing, and mission data processing and distribution. The Ground Station consists of the computers, network, communication equipment, and

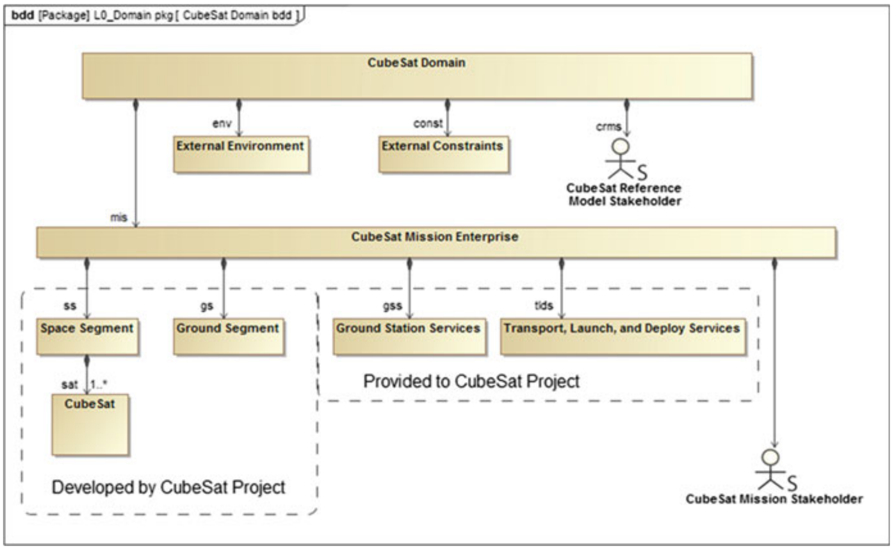

bdd [Package] L0_Domain pkg [CubeSat Domain bdd]
CubeSat Domain
env
const
crms
External Environment
External Constraints
CubeSat Reference Model Stakeholder
mis
CubeSat Mission Enterprise
ss
gs
gss
tlds
Space Segment
Ground Segment
Ground Station Services
Transport, Launch, and Deploy Services
Provided to CubeSat Project
sat
1..*
CubeSat
Developed by CubeSat Project
CubeSat Mission Stakeholder

Fig. 27.1 CubeSat Domain and Mission Enterprise

associated control infrastructure hosted in a ground facility. Communication equipment includes the space–ground antennas. The architecture accommodates a CubeSat project developing its own ground station or operating with an existing ground station that provides uplink and downlink services.

Subsystems Figures 27.2 and 27.3 show the decomposition into logical subsystems of both the Space and Ground Segments. While these subsystems currently comprise the logical partitioning of the segments, they may later reflect the physical partitioning as well. Starting with this list, teams may add or remove subsystems based on the mission requirements and objectives.

Model Organization. There are packages for the domain, enterprise, space and ground segments, and space and ground subsystems. The enterprise, segment, and subsystems packages contain behaviors, structures, validation, and verification packages.

Requirements Requirements are organized by enterprise, space and ground segments, and space and ground subsystems packages. The enterprise package consists of mission needs, mission objectives, mission constraints, and mission requirements packages with model elements to establish the relationships to the stakeholder needs, objectives, constraints, and measures of effectiveness. Figure 27.4 shows model elements that help define lower-level requirements. The relationships between elements are illustrative not prescriptive. Segment requirements are derived from mission requirements and trace to mission use cases. Segment requirements trace to measures of performance that trace to measures of effectiveness. Subsystem requirements are derived from segment requirements and trace to

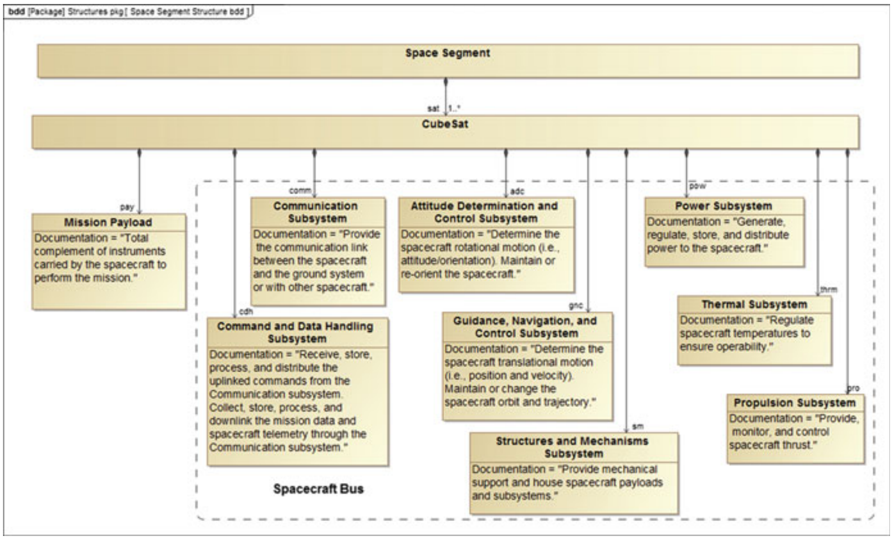

bdd [Package] Structures pkg [Space Segment Structure bdd]
Space Segment
sat 1..*
CubeSat
pay
Mission Payload
Documentation = "Total complement of instruments carried by the spacecraft to perform the mission."
comm
Communication Subsystem
Documentation = "Provide the communication link between the spacecraft and the ground system or with other spacecraft."
adc
Attitude Determination and Control Subsystem
Documentation = "Determine the spacecraft rotational motion (i.e., attitude/orientation). Maintain or re-orient the spacecraft."
pow
Power Subsystem
Documentation = "Generate, regulate, store, and distribute power to the spacecraft."
cdh
Command and Data Handling Subsystem
Documentation = "Receive, store, process, and distribute the uplinked commands from the Communication subsystem. Collect, store, process, and downlink the mission data and spacecraft telemetry through the Communication subsystem."
gnc
Guidance, Navigation, and Control Subsystem
Documentation = "Determine the spacecraft translational motion (i.e., position and velocity). Maintain or change the spacecraft orbit and trajectory."
thrm
Thermal Subsystem
Documentation = "Regulate spacecraft temperatures to ensure operability."
pro
Propulsion Subsystem
Documentation = "Provide, monitor, and control spacecraft thrust."
sm
Structures and Mechanisms Subsystem
Documentation = "Provide mechanical support and house spacecraft payloads and subsystems."
Spacecraft Bus

Fig. 27.2 CubeSat Space Segment

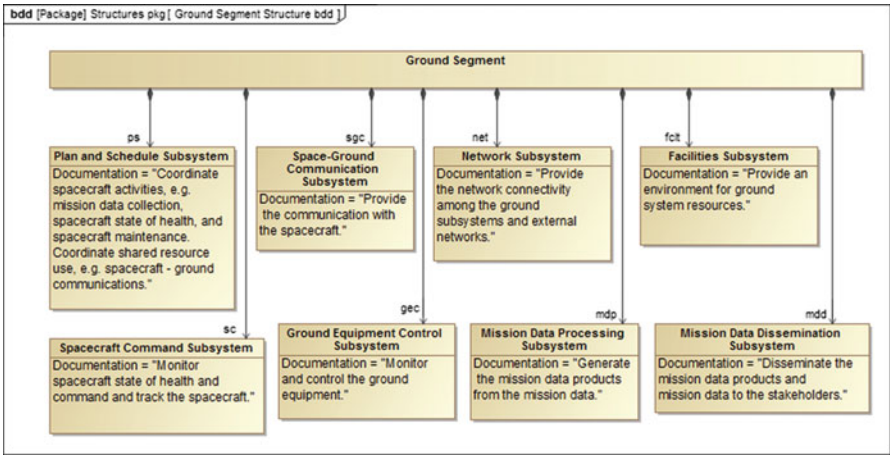

bdd [Package] Structures pkg [Ground Segment Structure bdd]
Ground Segment
ps
Plan and Schedule Subsystem
Documentation = "Coordinate spacecraft activities, e.g. mission data collection, spacecraft state of health, and spacecraft maintenance. Coordinate shared resource use, e.g. spacecraft - ground communications."
sgc
Space-Ground Communication Subsystem
Documentation = "Provide the communication with the spacecraft."
net
Network Subsystem
Documentation = "Provide the network connectivity among the ground subsystems and external networks."
fclt
Facilities Subsystem
Documentation = "Provide an environment for ground system resources."
sc
Spacecraft Command Subsystem
Documentation = "Monitor spacecraft state of health and command and track the spacecraft."
gec
Ground Equipment Control Subsystem
Documentation = "Monitor and control the ground equipment."
mdp
Mission Data Processing Subsystem
Documentation = "Generate the mission data products from the mission data."
mdd
Mission Data Dissemination Subsystem
Documentation = "Disseminate the mission data products and mission data to the stakeholders."

Fig. 27.3 CubeSat Ground Segment

segment use cases. Subsystem requirements trace to technical performance measures that trace to measures of performance and to measures of effectiveness.

Technical Measures. Technical measures provide the stakeholders insight into the definition and development of the technical solution. Measures of effectiveness, key performance parameters, measures of performance, and technical performance parameters are technical measures. They are distinct from requirements, although performance and other requirements may be traced to them. They are incorporated

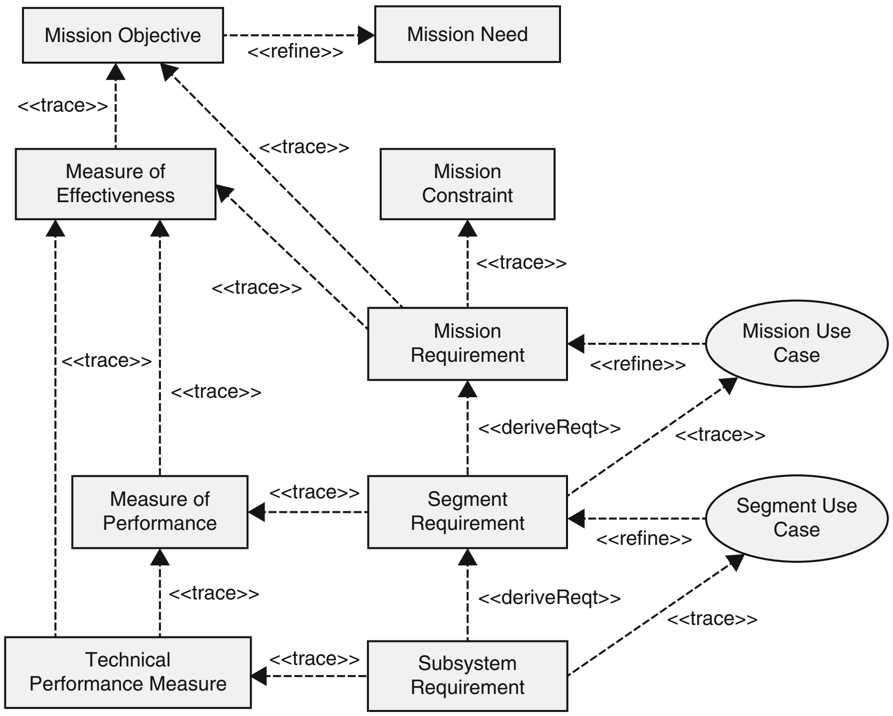


Fig. 27.4 Hierarchy of mission needs, objectives¸ and constraints; technical measures, requirements, and use cases. The relationships between elements are illustrative not prescriptive

into the CRM as block value properties. A technical measure can be measured and compared to a target value.

Functional and Nonfunctional Requirements Requirements can be classified as functional or nonfunctional. Functional requirements define the desired behaviors of the system and nonfunctional requirements define the overall qualities or attributes of the resulting system. Nonfunctional requirements (which include Quality Factors) place constraints on the product being developed, the development processes, or conformance with external regulations. Examples of nonfunctional requirements include safety, security, usability, reliability, and performance. However, as requirements are decomposed, the distinction between functional and nonfunctional requirements may disappear. For example, a top-level mission reliability requirement may lead to lower-level requirements that identify where redundancy would be needed. The presence of redundancy requires redundancy management functions whose behaviors could be considered functional requirements. Whether a requirement is expressed as a functional or nonfunctional requirement may depend on the level of detail to be included in the requirements document, the extent to which the application domain is understood, and the experience of the developers.

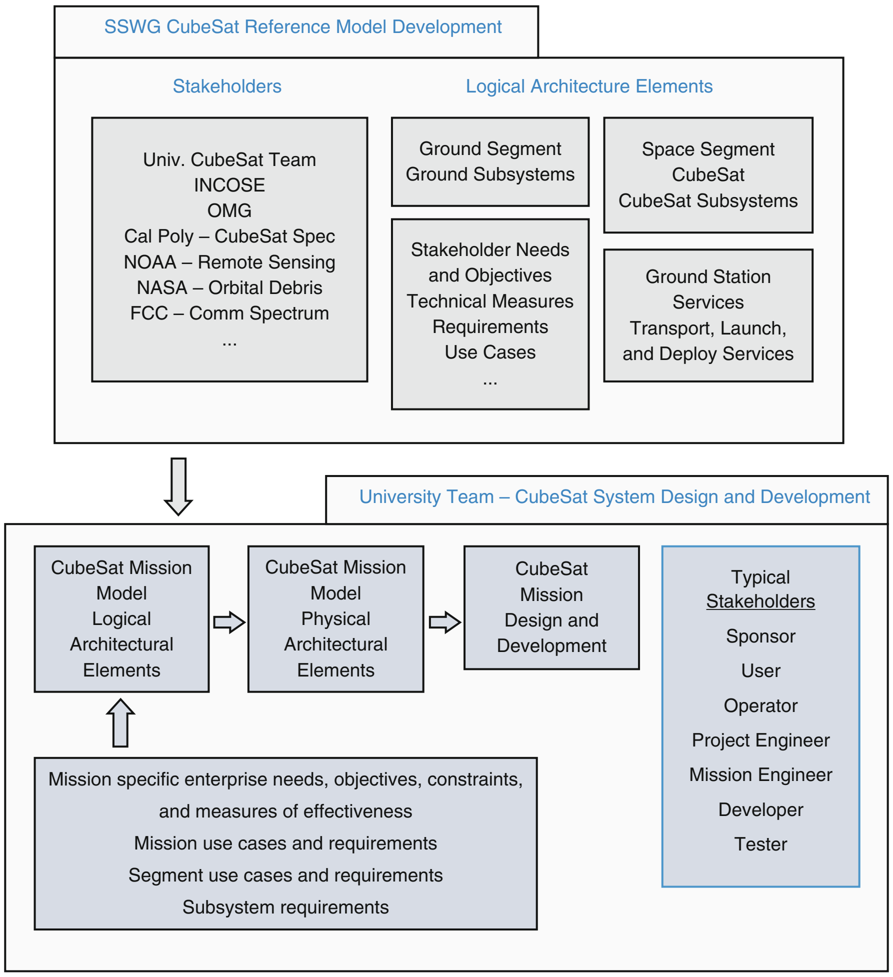

SSWG CubeSat Reference Model Development
Stakeholders
Logical Architecture Elements
Univ. CubeSat Team
INCOSE
OMG
Cal Poly – CubeSat Spec
NOAA – Remote Sensing
NASA – Orbital Debris
FCC – Comm Spectrum
...
Ground Segment
Ground Subsystems
Space Segment
CubeSat
CubeSat Subsystems
Stakeholder Needs and Objectives
Technical Measures
Requirements
Use Cases
...
Ground Station Services
Transport, Launch, and Deploy Services
University Team – CubeSat System Design and Development
CubeSat Mission Model Logical Architectural Elements
CubeSat Mission Model Physical Architectural Elements
CubeSat Mission Design and Development
Typical Stakeholders
Sponsor
User
Operator
Project Engineer
Mission Engineer
Developer
Tester
Mission specific enterprise needs, objectives, constraints, and measures of effectiveness
Mission use cases and requirements
Segment use cases and requirements
Subsystem requirements

Fig. 27.5 CubeSat reference model provides the foundation for the mission-specific CubeSat model

27.3 Mission-Specific CubeSat Model [13]

The steps for developing a mission-specific CubeSat model are illustrated in Fig. 27.5. The first step is taking the CRM and populating the mission-specific enterprise needs, objectives, constraints, and measures of effectiveness to create a mission-specific logical architecture. Figure 27.4 previously illustrated the roles and relationships of requirements, use cases, and technical measures across the architectural layers comprising the enterprise, mission, segments, and subsystems. Key to defining the mission-specific logical architecture layers is creating of use cases and technical measures to describe fully the behaviors and data flow in support of the stakeholder needs, objectives, measures of effectiveness, and constraints. Although the CRM space and ground subsystems have been broadly defined, the mission teams may find it necessary to modify the subsystem definitions according to the allocated requirements.

The next step is to create the physical architecture from the logical architecture, and this is accomplished by determining the types of subsystem components that meet the functional and performance subsystem requirements. Physical components include the specific hardware, software, persistent data, and operational procedures.

The final step in Fig. 27.5 is to complete the CubeSat mission design and to develop the CubeSat space and ground segments.

27.4 Approach to Validation and Verification [13]

The CRM is basically a model of a model. That is, the CRM will be used by a mission-specific CubeSat team to design and develop their mission-specific CubeSat.

Validation confirms, by providing objective evidence, that the system, as-built (or as it will be built), satisfies the stakeholders' needs. That is, the right system has been (or will be) built.

Verification confirms, by providing objective evidence, that the system and all its elements perform their intended functions and satisfy the requirements allocated to them. That is, the system has been built right. Verification methods include inspection, analysis, demonstration, and test.

Stakeholders are individuals or organizations with an interest in the system. Typical stakeholders include users, operators, organization decision makers, parties to the agreement, regulatory bodies, developing agencies, support organizations, and society at large. They can also include interoperating and enabling systems.

Stakeholders have various interests in the CRM: Some are interested in the models themselves and others are interested in the missions that can be realized from the mission-specific instantiations of the model, and some are interested in both.

Stakeholders, Concerns, Viewpoints, and Views ISO/IEC/IEEE 42010:2011 established the following terminology [15]:

- Stakeholders and Concerns: A concern could be manifest in many forms, such as in relation to one or more stakeholder needs, goals, expectations, responsibilities, requirements, design constraints, assumptions, dependencies, quality attributes, architecture decisions, risks or other issues pertaining to the system.
- Architecture Viewpoint: Work product establishing the conventions for the construction, interpretation and use of architecture views to frame specific system concerns.

- Architecture View: Work product expressing the architecture of a system from the perspective of specific system concerns.

Regulatory agencies are stakeholders. Licenses and regulations, timelines, and procedures must be must be well-understood and part of the CRM. In the USA, the FCC regulates the radio frequencies, NASA provides orbital debris guidelines, and NOAA regulates remote sensing. The validation that the national stakeholders' regulations and guidelines have been properly instantiated will consist of viewpoints into the CRM. The viewpoints include source regulations, guidelines, procedures, and timelines. Verification of compliance with the regulations and timelines will be the responsibility of the mission-specific CubeSat team. Their mission-specific CubeSat model will need viewpoints for the compliant model elements, licenses, and authorizations.

The Cal Poly CubeSat Project is a stakeholder. The Cal Poly CubeSat Specification [16] specifies a CubeSat's physical, mechanical, electrical, testing, and operational requirements.

The INCOSE and OMG are both stakeholders. They jointly developed SysML to support MBSE. An independent review team will validate that the CRM complies with accepted SysML modeling guidelines. OMG is responsible for establishing the CRM as a specification. OMG review and approval of the CRM will validate that the CRM is qualified to be a specification.

The SSWG and University CubeSat Teams are both stakeholders since they are model users. The SSWG is a stakeholder since it is developing the model for use by the university team. A traditional pre-MBSE approach would be to negotiate a CRM requirements document and then to develop the model. In MBSE, the SSWG works with the university teams to define the model elements and relationships from the CubeSat domain and enterprise to the space and ground segments and subsystems.

Figure 27.6 illustrates that viewpoints into the CRM will provide the objective evidence needed for validation. The CRM will be populated with a representative mission, and then the viewpoints will provide the objective evidence for verification. The CRM will have logical elements that can be reused by a mission-specific CubeSat team as a basis for its logical and physical CubeSat models. The CRM will have viewpoints for model elements and relationships in support of mission-specific CubeSat stakeholder needs, objectives, and technical elements as well as requirements definition, validation, and verification. As illustrated in Fig. 27.6, the mission-specific CubeSat viewpoints will provide the objective evidence needed for validation and verification of the mission-specific CubeSat model. Figure 27.6 also shows the role of mission modeling in the validation and verification of the mission-specific CubeSat model and the mission-specific CubeSat. The CubeSat SysML model and the modeling tool can be configured to execute a mission scenario. This includes interfacing with commercial off-the-shelf (COTS) modeling tools [7].

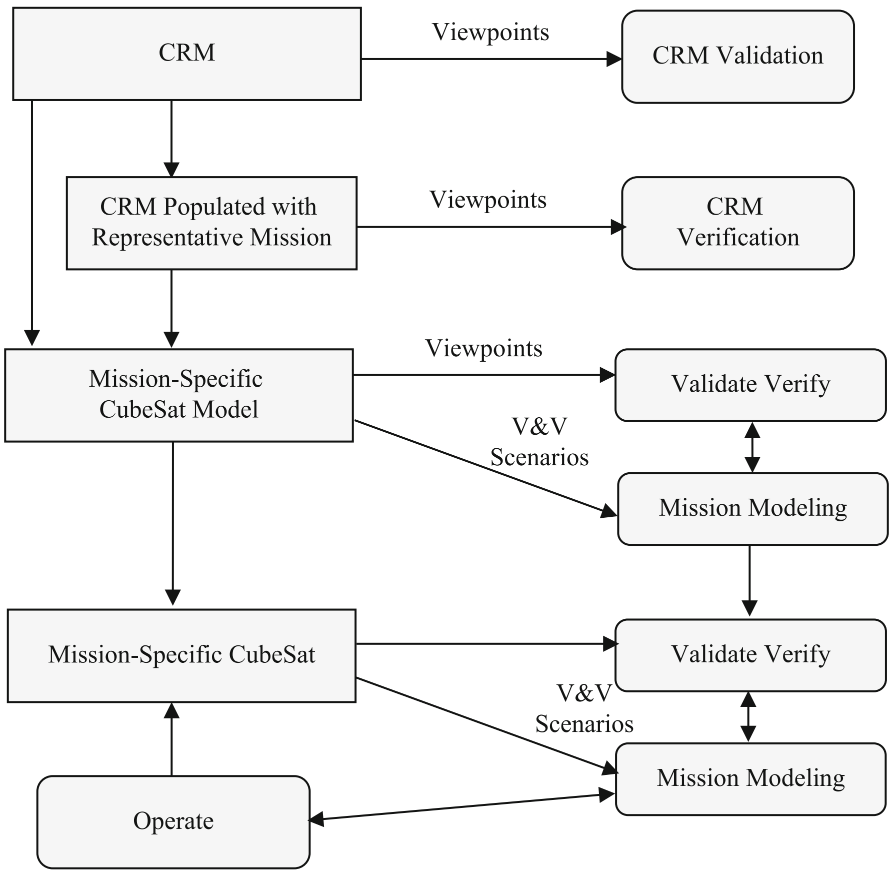


Fig. 27.6 Validation and verification of the CubeSat reference model and the mission-specific CubeSat model

27.5 Conclusion

As MBSE matures and continues to penetrate real-world engineering practice, the concept of a reference model is an important development to facilitate transition of MBSE into real-world practice [1, 17]. This paper has presented recent advances in developing a CubeSat Reference Model (CRM) in accord with MBSE specification. This paper discusses the specification, validation, and verification of the CRM. This activity is being pursued by a team comprising government, academia, and INCOSE SSWG. This activity is expected to continue into the foreseeable future resulting in further advances in MBSE [17, 18]. Future advances include the development of a library of reference models (for different domains and missions) that are metadata-tagged for easy retrieval.

Acknowledgments The authors acknowledge the contributions of SSWG team members Philip Cahill, Myron Hecht, and Brad Ayres in developing the Validation and Verification approaches expressed in this paper.

References

1. Madni AM, Sievers M (2016) Model based systems engineering: motivation, current status and needed advances. Accepted for publication in Systems Engineering, 2016
2. International Council on Systems Engineering (INCOSE) (2007) MBSE initiative, January 2007. [Online] Available: http://www.omgwiki.org/MBSE/doku.php
3. Madni AM (2009) Systems architecting (SAE 549). Lecture notes, systems architecting and engineering program, Viterbi School of Engineering, University of Southern California
4. Madni AM (2011) Model-based systems engineering (SAE 548). Lecture notes, systems architecting and engineering program, Viterbi School of Engineering, University of Southern California
5. Reference Model for Service Oriented Architecture 1.0 (2006) OASIS standard, 12 October 2006
6. Swartwout M (2016) University-class spacecraft by the numbers: success, failure, debris (but mostly success). 30th annual AIAA/USU conference on small satellites, logan UT, 2016. Approximately 75% of university-class missions were CubeSats, with a mission failure rate of some 50% (when launch failures were discounted)
7. Spangelo S, Cutler J, Anderson L, Fosse E, Cheng L, Yntema R, Bajaj M, Delp C, Cole B, Soremekun G, Kaslow D (2013) Model Based Systems Engineering (MBSE) applied to Radio Aurora Explorer (RAX) CubeSat mission operational scenarios. In: Proceedings of IEEE aerospace conference, Big Sky MT, University CubeSat design efforts have been largely based on intuition
8. Kaslow D, Soremekun G, Kim H, Spangelo S (2014) Integrated Model-Based Systems Engineering (MBSE) applied to the simulation of a CubeSat mission. In: Proceedings of IEEE aerospace conference, Big Sky, MT, March 2014
9. Kaslow D, Anderson L, Asundi S, Ayres B, Iwata C, Shiotani B, Thompson R (2015) Developing a CubeSat Model-Based System Engineering (MBSE) reference model – interim status. In: Proceedings of IEEE aerospace conference, Big Sky, MT, March 2015
10. Kaslow D, Ayres B, Chonoles M, Gasster S, Hart L, Massa C, Yntema R, Shiotani B (2016) Developing a CubeSat Model-Based System Engineering (MBSE) reference model – interim status #2. In: Proceedings of IEEE aerospace conference, Big Sky, MT, March 2016
11. Kaslow D, Ayres B, Chonoles M, Gasster S, Hart L, Levi A, Massa C, Yntema R, Shiotani B (2016) Developing and distributing a CubeSat Model-Based System Engineering (MBSE) reference model – status. In: Proceedings of 32 space symposium, Colorado Springs, April 2016
12. Kaslow D, Ayres B, Cahill P, Chonoles M, Hart L, Iwata C, Levi A, Yntema R (2016) CubeSat Model-Based Systems Engineering (MBSE) reference model – development and distribution – interim status. In: Proceedings of AIAA space forum, Pasadena, August 2016
13. Kaslow D, Ayres B, Cahill P, Hart L, Yntema R (2017) Developing a CubeSat Model-Based System Engineering (MBSE) reference model – interim status #3. In: Proceedings of IEEE aerospace conference, Big Sky, MT, March 2017
14. Kaslow D, Ayres B, Cahill P, Hart L, Yntema R (2017) A Model-Based Systems Engineering (MBSE) approach for defining the behaviors of CubeSats. In: Proceedings of IEEE aerospace conference, Big Sky, MT, March 2017
15. ISO/IEC/IEEE 42010:2011 Systems and software engineering – architecture description

16. CubeSat Design Specification (2014) Rev. 13, The CubeSat program, Cal Poly SLO, February 2014
17. Madni AM, Spraragen M, Madni CC (2014) Exploring and assessing complex system behavior through model-driven storytelling. In: IEEE systems, man and cybernetics international conference, invited special session "Frontiers of model based systems engineering", San Diego, 5–8 October 2014
18. Madni AM (2014) Generating novel options during systems architecting: psychological principles, systems thinking, and computer-based aiding. Syst Eng 17(1):1–9

Chapter 28
An Architecture Profile for Human–System Integration

Douglas W. Orellana and Azad M. Madni

Abstract Current architectural frameworks have adequate semantics to represent the hardware and software elements of a system. However, they are deficient when it comes to representing and integrating humans with the rest of the system. Current system engineering practices address human considerations as an afterthought. With advances in methods for human–system integration, there is an opportunity to extend current architecture modeling semantics to include humans and human–system integration semantics. To better integrate humans in and with systems, traditional systems engineering semantics need to be extended with human behavior representation models. In order to implement human considerations within complex system representations, this paper presents a Human–System Integration Profile, which was specifically created to represent human aspects as a lens within the overall system architecture. The Human–System Integration Profile is expected to be one of many tools that will be needed to fully integrate human considerations with system architectures and perform human–system trade-offs.

Keywords Model-based systems engineering • Human–system integration • SysML • Systems architecting • Systems architecture • Systems engineering • System modeling

28.1 Introduction

Today, with the role of the human changing from that of an operator to that of an agent [1, 2] and systems becoming increasingly more adaptable, greater demands are being placed on the system architect and engineer. Specifically, the human element needs to be taken into account and appropriately modeled from system conception to disposal. Current systems engineering practices address human–system integration as an afterthought (i.e., after architectures have been already specified and designed). In this situation, when changes to the system accumulate, redesign costs can spiral out of control. The key issue is that people not trained in

D.W. Orellana (✉) • A.M. Madni
University of Southern California, Los Angeles, CA, USA
e-mail: dworella@usc.edu; azad.madni@usc.edu

human factors engineering are unable to communicate with those that are, due to differences in terminology. To better integrate humans into systems, new semantics are needed to extend current system modeling semantics. The integration of the new semantics will allow for human elements to be analyzed in the holistic view of the system.

This paper will look at one way to better integrate human considerations into systems architecting, by extending current system modeling languages in order to integrate human–system integration (HSI) semantics. The profile is one step closer to fully integrating the human–system integration domain into everyday practice of the system engineer and architect.

28.2 Why Is an Architecture Profile Needed for Human–System Integration?

The objective of the HSI profile is to consider human actions as another set of perspectives in the system architecture. This multifold consideration of human actions is intended to increase the functional effectiveness and it should allow applying information about human characteristics and behavior into a more systematic way [3].

Within the system, human and machine trade-offs must be made. In the past, humans were usually modeled as external entities [4]. But in accordance with ISO 15288, humans are now treated as agents and must be considered as any other subsystem. Analyzing how the interactions between the subelements work as one. The human agent senses the outputs of the machine, and the human response is the input to the machine. Both machine and humans have a set of required capabilities and functionality to meet the system's goals and objectives.

Current system architecture tools do not take a holistic view on human considerations, which is often left for detailed design, well after the architecture is set, and major design decisions with huge monetary implications have been made. Like other engineering domains [5], standardized profiles are slowly beginning to get traction as new modeling semantics are needed to ease integrations of engineering methodologies, processes, and tools as well as opening up communications between various engineering disciplines.

Very minimal work has been done within the HSI community or system engineering community to integrate more human considerations into systems architecting. Current and past attempts [6, 7] have built partial constructs but neither has fully come to fruition and or extended use. IDEF 8 was centered in using interaction diagrams (activity-based diagrams) to allocate functions between a user and a system. The functions described had to deal with actions detailing interactions with physical controls and displays. Then with the use of the library of metaphors, the designers are able to use the metaphors to design the controls and displays of the system. Although IDEF 8 was a good attempt to bring up human–

system interaction upfront in the life cycle, it never took off and it had limited coverage of HSI issues. The Human Views propositioned for NATO Architectural Framework (NAF) is intended to document the unique implications of humans for system design from an acquisition perspective. The current views tend to focus on very high-level organizational considerations that allow for high-level trades of operational and enterprise considerations.

In order to fill the current gap, the Human–System Integration Profile attempts to align common HSI semantics per [8]. The HSI ontology looks at various areas within the framework of the system modeling pillars and other considerations that give a more holistic system view with the perspective of the human. Collectively, these factors provide the semantic underpinnings for defining and managing the human element within the mission and system context.

28.3 Human–System Integration Profile

Human–System Integration Profile is one implementation of the HSI Ontology [8] that is composed of mechanisms, requirements, human agents, behavior, structure, and parametrics. The Human–System Integration Profile focuses on developing the modeling constructs for requirements, human agents, behavior, and structure and is governed by the mechanisms identified in the ontology. The following subsections discuss in greater detail the various constructs that were developed to extend Unified Modeling Language (UML) based tools, in particular No Magic's Cameo EA.

Figure 28.1 shows the HSI Profile stereotypes. As shown in this figure, the profile stereotypes include various considerations associated with a human agent that need to be taken into account in the system architectural extensions. These considerations are the key to HSI. Table 28.1 presents the HSI concept extensions. The table specifically shows the HSI concept, the UML primitives, and the SysML categories associated with the concept.

Details of the SysML pillars are described in some detail next. Specifically, requirements, human agent, behavior, and structure are elaborated in the following paragraph.

28.3.1 Requirements Pillar

28.3.1.1 Human–System Interaction Requirement

Human–System Interaction Requirements are functional requirements that describe an interaction between a human agent and a machine. Requirements that will use this concept will describe a human agent function or task.

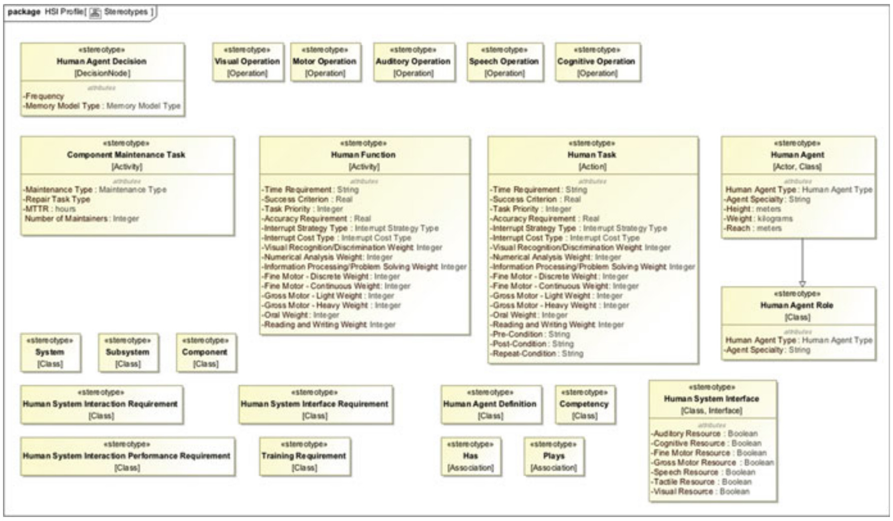

package HSI Profile[Stereotypes]
«stereotype» Human Agent Decision [DecisionNode]
attributes
-Frequency
-Memory Model Type : Memory Model Type
«stereotype» Visual Operation [Operation]
«stereotype» Motor Operation [Operation]
«stereotype» Auditory Operation [Operation]
«stereotype» Speech Operation [Operation]
«stereotype» Cognitive Operation [Operation]
«stereotype» Component Maintenance Task [Activity]
attributes
-Maintenance Type : Maintenance Type
-Repair Task Type
-MTTR : hours
Number of Maintainers : Integer
«stereotype» Human Function [Activity]
attributes
-Time Requirement : String
-Success Criterion : Real
-Task Priority : Integer
-Accuracy Requirement : Real
-Interrupt Strategy Type : Interrupt Strategy Type
-Interrupt Cost Type : Interrupt Cost Type
-Visual Recognition/Discrimination Weight Integer
-Numerical Analysis Weight : Integer
-Information Processing/Problem Solving Weight Integer
-Fine Motor - Discrete Weight : Integer
-Fine Motor - Continuous Weight : Integer
-Gross Motor - Light Weight : Integer
-Gross Motor - Heavy Weight : Integer
-Oral Weight : Integer
-Reading and Writing Weight Integer
«stereotype» Human Task [Action]
attributes
-Time Requirement : String
-Success Criterion : Real
-Task Priority : Integer
-Accuracy Requirement : Real
-Interrupt Strategy Type : Interrupt Strategy Type
-Interrupt Cost Type : Interrupt Cost Type
-Visual Recognition/Discrimination Weight Integer
-Numerical Analysis Weight : Integer
-Information Processing/Problem Solving Weight Integer
-Fine Motor - Discrete Weight : Integer
-Fine Motor - Continuous Weight : Integer
-Gross Motor - Light Weight : Integer
-Gross Motor - Heavy Weight : Integer
-Oral Weight : Integer
-Reading and Writing Weight Integer
-Pre-Condition : String
-Post-Condition : String
-Repeat-Condition : String
«stereotype» Human Agent [Actor, Class]
attributes
Human Agent Type : Human Agent Type
-Agent Specialty : String
-Height : meters
-Weight : kilograms
-Reach : meters
«stereotype» Human Agent Role [Class]
attributes
Human Agent Type : Human Agent Type
-Agent Specialty : String
«stereotype» System [Class]
«stereotype» Subsystem [Class]
«stereotype» Component [Class]
«stereotype» Human System Interaction Requirement [Class]
«stereotype» Human System Interface Requirement [Class]
«stereotype» Human Agent Definition [Class]
«stereotype» Competency [Class]
«stereotype» Human System Interface [Class, Interface]
attributes
-Auditory Resource : Boolean
-Cognitive Resource : Boolean
-Fine Motor Resource : Boolean
-Gross Motor Resource : Boolean
-Speech Resource : Boolean
-Tactile Resource : Boolean
-Visual Resource : Boolean
«stereotype» Human System Interaction Performance Requirement [Class]
«stereotype» Training Requirement [Class]
«stereotype» Has [Association]
«stereotype» Plays [Association]

Fig. 28.1 Human–System Integration Profile Stereotypes

28.3.1.2 Human–System Interaction Performance Requirement

Human–System Interaction Performance Requirements are performance requirements that describe performance of an interaction between a human agent and a machine. The interaction performance requirement is composed at a minimum of definition of interaction timing, accuracy, and success criteria. The timing requirement specifies the amount of time a human agent function or task needs to complete in. The accuracy requirement specifies the accuracy to which the human agent can complete a human function or task. The success criteria requirement specifies the percentage of time a human function or task is completed.

28.3.1.3 Human–System Interface Requirements

Human–System Interface Requirements specify sensory and physical characteristics that can affect the effectivity of the human–system interface. These requirements can be against a graphical user interface, virtual interfaces, and physical interfaces. Graphical user interfaces are usually designed using standards for colors and layout given the organization. The amount of data being presented can affect human reaction as well as the frequency of data update. Requirements to reduce human agent fatigue and keep human agent arousal are key to reducing the risk of error. By identifying human–system interface requirements, engineers would ensure that they are considered earlier on in system trades.

Table 28.1 Human–system integration concept extension

Concept	UML	SysML
Human–system interaction requirement	Class	Requirement
Human–system interaction performance requirement	Class	Requirement
Human–system Interface requirement	Class	Requirement
Human agent definition	Class	Requirement
Training requirement	Class	Requirement
Human agent	Class, actor	Block, actor
Role	Class	Block
Human task network	Activity diagram	Activity diagram
Human agent function	Activity	Activity
Human agent task	Action	Action
Component maintenance task	Action	Action
Human agent decision	Decision node	Decision node
Human Interface interaction diagram	Sequence diagram	Sequence diagram
Visual operation	Operation	Operation
Motor operation	Operation	Operation
Auditory operation	Operation	Operation
Speech operation	Operation	Operation
Cognitive operation	Operation	Operation
Human agent state machine	State machine diagram	State machine diagram
Human–system interface	Interface	Interface
Storyboards	Class diagram	Graphically capture what the human agent must do in order to interact with the machine portion of the system
Human–system interface mock-up diagram	Class diagram	A visual depiction of the graphical interfaces that the human agent will be interfacing with

28.3.1.4 Human Agent Definition

"You are only as strong as your weakest link." When it comes to having the human agent being part of the system, it is the one element that can be unpredictable and may be considered the weakest link. In order to optimize the system, the human agent needs to be defined through personal aptitude and skill level. The personal aptitude describes innate or acquired abilities that correspond to system operations. The skill level will define the level of expertise that the human agent may need in order to be able to complete allocated functions and tasks within the appropriate accuracy and success completion rate.

Other aspects of the human definition that can limit the system effectiveness are stressors, either through the system environment (temperature, humidity, etc.) or physical (hours of sleep, hours on the job, etc.). By identifying stressors early on you can make decisions on how to reduce the stressors either through functional allocation or environmental controls. Fatigue can also be attributed to natural limit for work; knowing the overload threshold for the human agent can allow engineers to make some decisions on how many agents are needed or if there are other ways to reduce load.

28.3.1.5 Training Requirement

Training requirements specify the needed training a human agent would be required to have in order to be part of the system. This may include the frequency of the training and the standards for evaluating how effective the human agent is as well as the training itself. Training requirements would detail knowledge areas and skills areas for a human agent as well as the training methods that may be needed to reach the goal of preparing the human agent to those specifications.

28.3.2 Human Agent Pillar

28.3.2.1 Human Agent

The human agent concept represents the human element of the system. As part of the human element definition, a minimum set of attributes define limitations of the human agent: skill level, specialty, training frequency, and interface usage.

The skill level attribute is a measure of whether the agent is a novice or a journeyman. The skill level is related to the given specialty the agent is assigned. The specialty attribute is an assigned mission role the agent should be serving. For each specialty, the human agent should keep a training frequency that will attribute to the skill level the human agent has.

Due to the fact that the human agent must interface with a machine in some manner, in order for the human to be more efficient, it is necessary to know how often the human agent uses the interface. The amount of usage of the interface will also attribute to the skill level the human agent will have.

28.3.2.2 Role

A Role is a type of agent that the system needs for its operation. The difference between the role and the human agent is that a human agent could serve many roles, although in an ideal world to reduce the amount of context switching one role will be played by one human agent with the same specialty for better efficiency. A Role is similarly defined like a Human Agent with specialty, skill level, training frequency, and interface usage attributes.

Some highly used roles used in HSI analyses are the Operator, Maintainer, and Supply and Support Personnel. The Operator role uses a system and typically is defined by the mission-owning organization. The Maintainer role maintains a system. The Supply and Support Personnel is typically related to defense systems and are in charge of ensuring systems are supplied with the necessary equipment to complete the mission.

28.3.3 Behavior Pillar

28.3.3.1 Storyboards

Storyboards are an ideal platform for capturing human agent stories. They graphically capture what the human agent must do in order to interact with the machine portion of the system.

28.3.3.2 Human Task Network

Human Task Network captures the decomposition of human agent functions into lower-level human agent function and human agent tasks. The functions/tasks are structured to show the flow between the lower-level functions and tasks. Human task networks can define aspects of the storyboard.

28.3.3.3 Human Agent Function

A Human Agent Function relates to an operational activity as part of an operational activity model. The human agent function describes the human agent behavior in context of the operational activity. In most cases, human agent functions can be decomposed into human agent tasks and lower-level human agent functions. The flow of these set of decomposition functions/tasks is captured in a human task network.

28.3.3.4 Human Agent Task

A human task is the lowest level of behavior that a human agent completes. A human task consumes human agent resources and is used to complete an interaction with a human–system interface. The human agent task contributes to the workload a human agent experiences as well as contributes to the stressors the human agent experiences.

28.3.3.5 Component Maintenance Task

A Component Maintenance Task is a type of Human Agent Task that is specifically focused on maintaining a system. Some common maintenance tasks include adjust and repair, inspection, remove and replace, test and check, and troubleshoot. These maintenance tasks have attributes that describe the maintenance type, support level, maintainer specialty, maintenance skill level, number of maintainers, and mean time to repair.

The Component Maintenance Task can be preventative, scheduled, or corrective maintenance type. A Component Maintenance Task support level corresponds to the location where maintenance can be done. Organizational support maintenance can be done at the unit level and they are done on the system itself. Direct support is maintenance that needs to be done at a local facility. Maintenance can be done from unit level all the way to the system level at these direct support facility. General support is remotely done and only can be done at unit levels. Contact team support brings maintenance to the system. Each component maintenance task can be done by a human agent specialty with a specific skill level. In some cases, more than one maintainer is needed to complete component maintenance tasks. Like human agent tasks, component maintenance tasks have a time to complete, but in this case to maintain, mean time to repair (MTTR).

28.3.3.6 Human Agent Decision

In order to specify human decision-making the profile can identify occasions where the human agent must make a decision. For human interaction analysis, it is necessary to understand how often a human decision is being made and the type of memory it needs (working, short-term, and long-term memory). The type of memory being used may affect timing constraints as well as allow you to make adjustments on whether a human agent should be making the decision or if you should automate the decision.

28.3.3.7 Human–Interface Interaction Diagram

The human–interface interaction diagram focuses on the human agent interaction with the human system interface. Each operation described in this diagram describes a human resource that is being used in the use of the interface for a scenario.

28.3.3.8 Visual Operation

A visual operation is a visual function that must be done by a human agent to interact with a human–system interface. This can be defined in an interaction diagram describing lower-level interaction between the human agent and the human–system interface.

28.3.3.9 Motor Operation

A motor operation is a motor function that must be done by a human agent to interact with a human–system interface. Motor functions can be tactile, fine, or gross motor functions. This can be defined in an interaction diagram describing lower-level interaction between the human agent and the human–system interface.

28.3.3.10 Auditory Operation

An auditory operation is an auditory function that must be done by a human agent to interact with a human–system interface. This can be defined in an interaction diagram describing lower-level interaction between the human agent and the human–system interface.

28.3.3.11 Speech Operation

A speech operation is a speech function that must be done by a human agent to interact with another human agent. This can be defined in an interaction diagram describing lower-level interaction between human agents.

28.3.3.12 Cognitive Operation

A cognitive operation is a cognitive function that must be done by a human agent to interact with a human–system interface. This can be defined in an interaction diagram describing lower-level interaction between the human agent and the human–system interface.

28.3.3.13 Human Agent State Machine

The Human Agent State Machine attempts to capture the dynamic state of the human agent in accordance with conditions of the system and the environment.

28.3.4 Structure

28.3.4.1 Human–System Interface

A Human–System Interface is part of a subsystem that is part of a system. This interface is what incorporates the human agent into the system. Human–system interfaces can be physical and/or digital. Physical interface can include levers, buttons, etc. Digital interfaces are usually through a display and require a graphical user interface. The driving aspect of designing a human–system interface is the agent experience with the interface. If the interface is too cumbersome, the human agent will not be able to stay engaged for system success. System success is driven on how well these interfaces are designed.

28.3.4.2 Human–System Interface Mock-Up

In order to help with the success of graphical interfaces, the Human–System Interface Mock-Up is a visual depiction of the graphical interfaces that the human agent will be interfacing with. These mock-ups can be used in combination with storyboards to better understand the human–machine interactions through the human–system interface, decreasing uncertainty of interface use.

28.4 Benefits for Implementing a Human–System Integration Profile

Implementing the Human–System Integration Profile enables the introduction and use of human-related semantics within complex systems engineering semantics. By extending UML, SysML or both, architects can still build their architecture in the same way that they have in the past, but with the added benefit of highlighting important human considerations. Not only does the profile integrate to UML and SysML, but it also gives the architect the same look and feel that architects are accustomed to when using UML and SysML.

The stereotypes discussed in Sect. 28.3, not only add new object semantics, but also new diagrams: Storyboard, Human Task Network, Human–Interface Interaction Diagram, Human State Machine, and Human–System Interface Mock-Up. Each of these new diagrams gives one more lens into the architecture and complements the common used diagrams, peeling the level of complexity so better decisions can be made during development.

One of the reasons to create a Human–System Integration Profile was to better integrate descriptive models with human considerations with human–system integration analytical models. By integrating the human considerations, now architecture can be tightly analyzed for human agent workload through easier transitions to tools like Improved Research Integration Tool (IMPRINT) [9, 10].

28.5 Summary and Next Steps

This paper presented a Human–System Integration Profile that extends UML and SysML, standard-modeling languages for describing system architectures. Human considerations are now necessary since the roles of humans are no longer just of an operator, but are integral part of the overall system. The profile is an initial step to better integrate human considerations into the system architecting process. It will reduce the rework usually done when human considerations are left to the detailed design by moving those considerations to the left and as early as possible in the development process. By moving these considerations to the left, human–system integration trades can be done earlier in the process before decisions become very costly. As new human–system integration semantics are needed, the Human–System Integration Profile can be refined and added to keep in step with what is necessary to better consider the human within the system.

Within this area of research, the Human–System Integration Profile is only one piece of a suite of tools that will be needed to better analyze the system utilization and effectivity. One of the next steps is to use this implementation to allow for model interoperability between the architecture tool and the human–system integration analytical tool. In particular, integration information captured by UML-based tools can host the profile and use that information to populate and use that one source of truth to develop analyses within IMPRINT.

In current research agenda, a converter tool between a No Magic Cameo EA and IMPRINT is being developed that utilizes the Human– System Integration Profile. This converter tool will open up the possibilities to couple HSI analyses and the architecture to investigate how the human agent affects the system and its architecture. Whether the HSI analyses include workload models or maintenance models, these considerations will close the gap between the systems architecting process and the human–system design process. As the HSI ontology evolves and includes other HSI aspects (e.g., human failure and risk assessment), the HSI profile can include those semantics and add the number of analytical models that have direct integration with the architecture model.

References

1. Madni AM (2010) Integrating humans with software and systems: technical challenges and a research agenda. Syst Eng 13(3):232–245
2. Madni AM (2011) Integrating humans with and within complex systems. CrossTalk May/June:4–8
3. Balakrishnan PK (2002) Analysis of human factors in specific aspects of system design. In: INCOSE international symposium, pp 1–9
4. Arnold S, Earthy J, Sherwood-Jones B (2002) Addressing the people problem – ISO/IEC 15288 and the human-system life cycle. In: INCOSE international symposium, pp 1–7
5. Object Management Group (2013) UML testing profile v1.2

6. Mayer RJ, Crump JW, Fernandes R, Keen A, Painter MK (1995) Information integration for concurrent engineering (IICE) Compedum of methods report. Wright-Patterson Air Force Base, Ohio
7. Handley HAH, Smillie RJ (2008) Architecture framework human view: the NATO approach. J Syst Eng 11(2):156–164
8. Orellana DW, Madni AM (2014) Human system integration ontology: enhancing model based systems engineering to evaluate human-system performance. Procedia Comput Sci 28 (CSER):19–25
9. Wojciechowski JQ (2004) Validation of improved research integration tool (IMPRINT) driving model for workload analysis. Aberdeen Proving Ground. Army Research Lab (Aberdeen, MD)
10. Allender L (2000) Modeling human performance: impacting system design, performance, and cost. In: Proceedings of the military, government and aerospace simulation symposium, advanced simulation technologies conference, vol 32(3), pp 139–144

Chapter 29
Formal Methods in Resilient Systems Design: Application to Multi-UAV System-of-Systems Control

Azad M. Madni, Michael W. Sievers, James Humann, Edwin Ordoukhanian, Barry Boehm, and Scott Lucero

Abstract Resilience approaches today rely on ad hoc methods that offer piecemeal solutions. Models used by these methods are difficult to verify and do not scale. Furthermore, it is difficult to assess their long-term impact. This chapter presents a resilient systems design approach based on formal methods that is intended to overcome these limitations. The approach combines deterministic and probabilistic modeling to create a new modeling construct that lends itself to designing scalable, resilient systems and system-of-systems (SoS). The formalism facilitates model verification and possesses requisite flexibility to handle nondeterminism. The target application domain is multi-UAV swarm control in uncertain, potentially hazardous, dynamic environments. However, the approach is sufficiently general for a variety of SoS including autonomous vehicle SoS networks.

Keywords Resilient systems • Formal methods • Contract-based design • Partially observable Markov decision process

29.1 Introduction

Engineered resilience is a system property that allows a system to continue to provide useful service in the face of disruptions [1]. Disruptions can be external, systemic, or human-triggered. Disruptions within the context of multi-UAV swarms can range from hacked or compromised swarm member, loss of communication within the swarm or between specific swarm members, and loss of visibility due to extreme weather or sensor malfunction. Resilient responses to such

A.M. Madni (✉) • M.W. Sievers • J. Humann • E. Ordoukhanian • B. Boehm
University of Southern California, Los Angeles, CA, USA
e-mail: Azad.Madni@usc.edu

S. Lucero
Office of the Secretary of Defense, Washington, DC, USA
e-mail: Don.S.Lucero.civ@mail.mil

A.M. Madni et al. (eds.), *Disciplinary Convergence in Systems Engineering Research*, https://doi.org/10.1007/978-3-319-62217-0_29

disruptions can take a variety of forms: anticipate and circumvent disruptions; withstand disruptions within the designed system performance envelope; recover rapidly from the negative effects of disruptions; extend system capacity dynamically to cope with disruptions; restructure or reconfigure pursuant to disruptions; and continue to operate at some diminished level when disruptions fall outside the system's designed performance envelope. The system's design envelope includes system models and adaption mechanisms incorporated within the system model to produce resilient responses in the face of disruptions. The key distinction to be made here is between mechanisms in the system to respond in resilient fashion to uncertainties specifically addressed in the system model, and the recognition of contexts in which resilient responses require behaviors that fall outside the system design envelope. The latter is called unanticipated variability, disruption, and perturbation [2–4]. Doyle (2016) defines resilience as "the ability to recognize unanticipated perturbations that fall outside the purview of the system model designed to help the system adapt to disruptions that lie outside the system's design envelope" [5]. This implies that resilience is concerned with monitoring the boundary conditions of the system's model for competence (how well resilience strategies match disruption demands), and then adjusting or expanding that model to better accommodate changing demands. The key issue here is assessing the organization's adaptive capacity (i.e., resource buffers that allow resources of a particular type to be increased on demand to a maximum limit) relative to the challenge posed by the disrupting event to that adaptive capacity. The organization in this case is the multi-UAV swarm enterprise. Boundaries in this context are properties of the model that define the system's competent performance envelope relative to specific classes of disruptions and uncertainties. Thus, resilience engineering in some sense is concerned with introducing transparency into an organization's model of creating safety, with the express purpose of determining when the model needs to be revised. Thus, resilience engineering is concerned with monitoring an organization's decision making with a view to assessing an organization's risks and risk envelope relative to unsafe operation boundary.

Thus, resilience engineering is concerned with monitoring an organization's risks and risk envelope relative to unsafe operation boundary. The intent of risk monitoring is to proactively and automatically/semiautomatically monitor buffers, margins, and tolerances. Buffer capacity is concerned with the magnitude and type of disruptions a system can absorb or adapt to without a substantial degradation in system performance, or breakdown in integrity of system structure. Flexibility is the ability of a system to restructure or reorganize itself to effectively respond to external changes or pressures [6]. Margin is the proximity of a system's operation regime relative to its designed operational performance envelope or boundary. Tolerance is the ability of a system to degrade gracefully (as opposed to collapsing) as stress/pressure increases, or when disruption magnitude and/or severity exceeds its adaptive capacity.

This chapter presents a formal model-based approach to engineering and verifying resilient system designs. The chapter is organized as follows. Section 29.2 discusses organizational challenges in engineering resilient systems. Section 29.3

provides a detailed discussion of the formal modeling approach. Section 29.4 provides an illustrative example concerned with resilient control of multi-UAV swarms. Section 29.5 presents a discussion of future prospects for engineered resilient systems.

29.2 Technical Challenges

There are several technical challenges that have to be overcome in developing formal methods for the engineering of resilient systems including choosing the right system modeling construct, the right technology platform for development and demonstration, and the right application domain. These considerations are discussed next.

Application Domain We chose UAV swarm control as our application domain. A UAV swarm is a system-of-systems (SoS) in which the elements can be either homogeneous or heterogeneous. The elements in the SoS cooperate to perform their assigned mission or mutually agreed to tasks, and coordinate as needed. Each UAV in the swarm is equipped with sensors and communication facilities. UAV swarms are used in a variety of missions in the military and civilian sector. Exemplar missions include search and rescue, reconnaissance and surveillance, humanitarian assistance, and disaster relief.

System Modeling Construct Selecting the right modeling construct is a key challenge. The model needs to be semantically expressive, scalable, amenable to verification, and sufficiently flexible to support mechanisms needed to handle nondeterminism.

Technology Platform The technology platform for this effort needs to support SoS specification and visualization, deterministic and probabilistic modeling, and integration with analytics and reporting modules.

29.3 Formal Modeling of Systems and SoS

Formal modeling is a means to introduce rigor in system representation and reasoning. Formal models lend themselves to model verification and, potentially, model testing. However, formal modeling has limitations. The rigor in formal modeling comes at the expense of flexibility. Ideally, one wants sufficient formality to support model verification and testing, and sufficient flexibility to scale and cope with uncertainty. This recognition provided the impetus for the hybrid approach presented in this chapter.

Our approach extends contract-based design (CBD) to successfully address uncertainty and partial observability that contribute to nondeterministic behavior

[7, 8]. CBD is a formal method for explicitly defining, verifying, and validating system requirements, constructs, and interfaces. An implementation is considered to satisfy a design contract if it fulfills guarantees when assumptions are true. This is the assert-guarantee construct used in CBD. The rationale for choosing CBD is that statements in contracts are mathematically provable. The limitation of CBD is that the assertions are invariant. The key innovation in our approach is the relaxation of invariant assertions requirement to introduce flexibility in the contract. The resulting construct is a "resilience contract" (RC). The RC combines CBD and partially observable Markov decision process (POMDP). The rationale for POMDP is that it introduces flexibility into a traditional contract by allowing incomplete specification of legal inputs and flexible definition of postcondition corrections [7, 8].

A RC is a hybrid modeling construct (CBD + POMDP) in which the assertions associated with contract are flexible, and techniques employed include in-use learning, uncertainty handling, and pattern recognition. A RC, in essence, extends the deterministic contract to represent stochastic systems. A RC is developed at design time and trained during system use (learning). A RC is amenable to trading of model verification for model flexibility. The latter is needed for addressing model (system) resilience.

The rationale for POMDP is that many real-world problem situations are not fully observable, however, the Markovian assumptions is often valid. A POMDP model consists of a set of states' S, set of actions A, a set of observation O, a transition model, an observation model, and a reward model. The Markov assumption applies to the transition model, with the optimal policy depending only on the current state. In a partially observable environment, the current state is generally not known and, therefore, the agent cannot execute the optimal policy for that state. The POMDP model offers the basis for introducing flexibility in CBD. Specifically, flexibility can be introduced by first relaxing the time invariance restriction on the state space and action space, adding an evaluation metric to determine best action, updating the emission and transition probabilities of hidden states, and finally adding the concept of time. A key insight in introducing flexibility in contract is replacing the "assert-guarantee" construct with "belief-reward" construct. We hypothesized that this change is the basis for incorporating flexibility into contracts without compromising model verification and test benefits of CBD to any appreciable degree.

29.4 UAV Swarm Control Architecture

Figure 29.1 presents swarm control architecture based on creating an optimal policy based on belief estimates provided by the state estimator. The state estimator relies on observations from the UAV swarm, environment sensors, and Markov Decision Process (MDP) belief model to generate updated belief estimates. Policy actions act on the UAV swarm and are used by the state estimator to update state information.

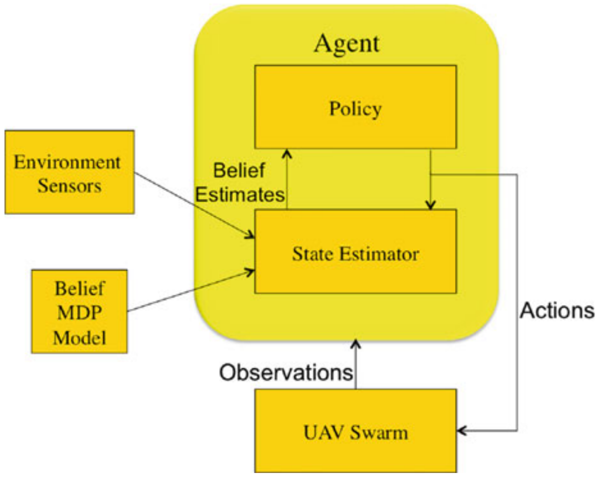


Fig. 29.1 Example swarm control architecture

A simple Concept of Operations (CONOPs) for the UAV swarm illustrates the approach. The UAV swarm needs to turn either left or right to avoid an obstacle. There is uncertainty regarding the location of the threat. The threat could be to the left or the right of the swarm. A decision needs to be made to veer left or veer right. If the swarm veers right and the threat is located/headed to the right, serious consequences could ensue. The same is true if the swarm veers left and the threat is heading left. There are three possible actions that the swarm can take: veer left; veer right; and continue flying straight ahead and collect more observations on the threat. POMDP policy for this simple CONOPS has to deal with a variety of considerations such as: UAVs not crash into each other; all UAVs get safely to their destination; UAVs avoid potentially disruptive events; if one or more UAVs is shot down, the remaining UAVs need to reorganize and reallocate functionality to ensure achievement of objective to the extent feasible. The key ideas behind an optimal POMDP policy are twofold: a POMDP policy maps current belief into an action and an optimal POMDP policy is a continuous solution of a belief MDP. Figure 29.2 shows the equation for summation of outcomes based on the path the UAVs take. The equation normalizes the rewards and penalties. As shown in Fig. 29.2, the system starts with a 50-50 belief that the threat could be to the left or the right. The system makes an observation. The system notices a potential threat to the left. So, the system moves its belief to the left as shown in the figure. That is, there is a greater belief that the threat could be to the left. Also, the system does not observe anything to the right. Thus, belief is updated in accord with Bayesian analysis using observation and current state.

One of the key problems with such state-space models is that they are subject to combinatorial explosion. To contain this explosion, several methods such as pruning (Bellman equation), branch and bound, heuristic search, Monte Carlo search, and policy tree can be applied.

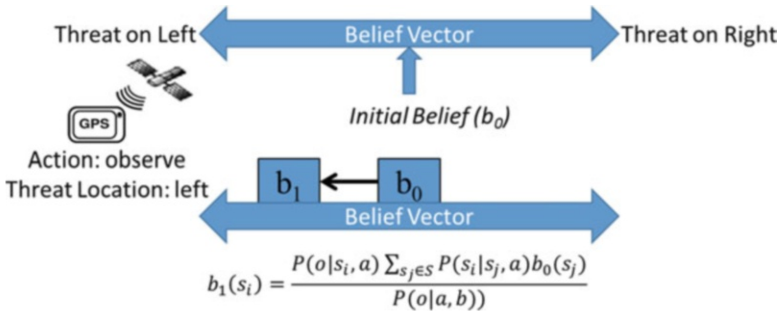


Fig. 29.2 Iterative update of beliefs

29.5 UAV Swarm Modeling

Our target application is the design resilient UAV swarms, and more specifically quadcopters. As noted earlier, the UAV swarm can be homogenous or heterogeneous. To model and evaluate system and SOS resilience, the first question that needs to be answered pertaining to model is fidelity and flexibility. Fidelity pertains to the depth of modeling and the perspectives model to successfully answer the questions posed. Flexibility pertains to the ease of extending or augmenting the model with mechanisms that introduce various forms of resilience. To this end, we need just enough fidelity and requisite flexibility. At the single UAV level, just enough flexibility requires capturing the nonlinear dynamics of the UAV (i.e., quadcopter) and taking into account aerodynamic effects. It also requires a basic sensor model and basic collision avoidance algorithm. The model needs to support both waypoint navigation and specific trajectory following. And, finally, the model should be easy to replicate in support of SoS requirements.

At the SoS level, the model needs to support different communication protocols, different missions, and different SoS configurations. The model should be capable of reflecting the behavior of hacked or compromised UAV in the SoS, loss of communication, loss of a UAV, loss of sensing, and malfunctioning SoS member. At both the individual UAV level and the swarm level, it should be possible to introduce and “test-drive” resilience concept*s*.

Behavioral Patterns and Use Cases UAV swarm behaviors can be conveniently grouped into four behavior patterns: deployment; en route; actions on objective; and redeployment. Each behavior pattern is discussed next.

Deployment (or Takeoff) Pattern Act of putting SoS into operation. UAVs initiate operations and take flight. Variations in pattern come in the form of takeoff method: vertical (VTOL), horizontal or conventional (CTOL), assisted (mechanical or human catapult, piggybacked from aircraft, propulsion assistance for short takeoff), etc; takeoff order: sequential versus parallel; swarm size, hierarchy, and homogeneity; mission: new, clean sheet deployment, or are UAVs reinforcing another UAV swarm; and platform: airfield, airport, grass field, naval ship, and improvised (such as a road or building top). The key factors affecting operation are characterized by Mission-Enemy-Troops-Terrain-and-Weather-Time Available-Civilian (METT-TC). An example of METT-TC factor is "enemy has robust air-defense in area necessitating unique flight maneuvers on takeoff."

En Route (or Cruise) Pattern Act of deployed swarm flying from one location to another in pursuit of overall mission. UAV SoS objective is navigate as appropriate in support of global mission, pathfind at a local level, maneuver through terrain, weather, other UAVs in SoS, and neighboring systems not a part of SoS (e.g., coalition aircraft, enemy aircraft, noncombatant aircraft), as well as making trade-offs in pathfinding and navigating in light of METT-TC. Variations in pattern come in the form of tactical flight considerations; high altitude versus midaltitude versus nap of the earth versus a combination; formation and disposition during cruise; swarm size, composition, and capabilities (swarm heterogeneity factors); enemy air defense capabilities and presence; and weather.

Actions on Objective Pattern Key part of overall CONOPS. Swarm achieves commander's intent and mission purpose. For example, reconnaissance, observation, sensing, collecting, aerial communications retransmit, kinetic: destroy enemy assets; neutralize enemy unit. UAV SoS objectives can be tactically pathfind at a local level both as individual systems and as a swarm to successfully execute actions on objectives, and deploy UAV systems as a SoS to achieve desired tactical and operational objectives in the battlespace. Variations in pattern – highly METT-TC dependent, for example, coordinated payload delivery to destroy a bridge and conduct recon; battlefield sensing and communications retransmission to support a focused, ground-based operation; routine mapping and imagery collection; search and rescue operation to locate downed aircraft in suspected geographical "crash window."

Redeployment Pattern Act of safely taking SoS out of operation. UAVs must RTB (return to base) and land, while preserving themselves and collected data (if held onboard). Variations of pattern come in the form of landing method: vertical (VTOL), horizontal or conventional (CTOL), assisted (tail hook and cable, parachute landing, or drag chute once landed); landing order: sequential versus parallel, swarm size, hierarchy, and homogeneity; mission: new, clean sheet deployment or are UAVs reinforcing another UAV swarm; platform: airfield, airport, grass field, naval ship, improvised (e.g., a road or building top); other METT-TC factors (e.g., enemy has robust air defense in area necessitating unique flight maneuvers on landing), and hasty landing (e.g., a damaged UAV improvises and lands in a clear area and sends out a distress signal Fig. 29.3).

Each basic pattern can be adapted and be decomposed into multiple more nuanced, specific scenarios using various METT-TC considerations for the SoS application. Fundamental concepts for top layer patterns are adapted and developed for highly specific use cases (e.g., fundamentals of an attack apply, but tactics behind attacking an enemy tank column vary – in the open vs. enemy ground troops in wooded mountains). Right level of decomposition and detail for each top-level

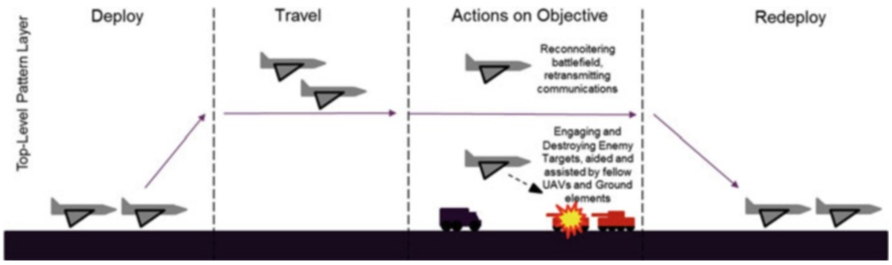

Top-Level Pattern Layer
Deploy
Travel
Actions on Objective
Redeploy
Reconnoitering battlefield, retransmitting communications
Engaging and Destroying Enemy Targets, aided and assisted by fellow UAVs and Ground elements

Fig. 29.3 UAV SoS CONOPS

pattern help answer questions about how best to incorporate resilience within the SoS.

Figure 29.4 shows the state transition diagram for a quadcopter. In this figure, some transitions are labeled with belief values, for example, b (failed) $\geq$0.95 is threshold of transition from "normal motors" to "failed motor," that is, transition happens if belief $\geq$0.95 that a motor has failed. Some transitions have fixed assertions, for example, failed motor and operational transition from "evaluate environment" to "auto plan enabled" has three beliefs with different probabilities in our example. Auto planner determines the course of action to take based on environment beliefs, motor condition beliefs, and goals (action taken is the one that maximizes reward or minimizes penalty).

A UAV swarm can be viewed as a SoS because multiple UAVs must work together to accomplish an end-to-end mission. A UAV swarm, especially a heterogeneous swarm, exhibits the characteristics of a SoS (Table 29.1).

29.6 Modeling Construct, Implementation, and Simulation

Our UAV modeling is based on the "sense-plan-act" construct (Fig. 29.5). The sensing and acting functions interact with the environment. The interactions consist of influencing the environment, and responding to events and changes in the environment.

Current implementation of UAV (quadcopter) encompasses an autopilot that accepts the desired trajectory as inputs as well as position and altitude feedback from quadcopter, and obstacle location coordinates from the obstacle detection system equipped with sensors. The quadcopter model receives control inputs from the autopilot and produces position state information as feedback to the autopilot and the obstacle detection system. The obstacle detection system accepts quadcopters current state information (i.e., position and altitude) and produces obstacle location information that is used by the autopilot. Figure 29.6 shows the current implementation of a single UAV control system.

The quadcopter architecture is shown in Fig. 29.7. The architecture captures quadcopter physical properties that drive quadcopter dynamics, desired/

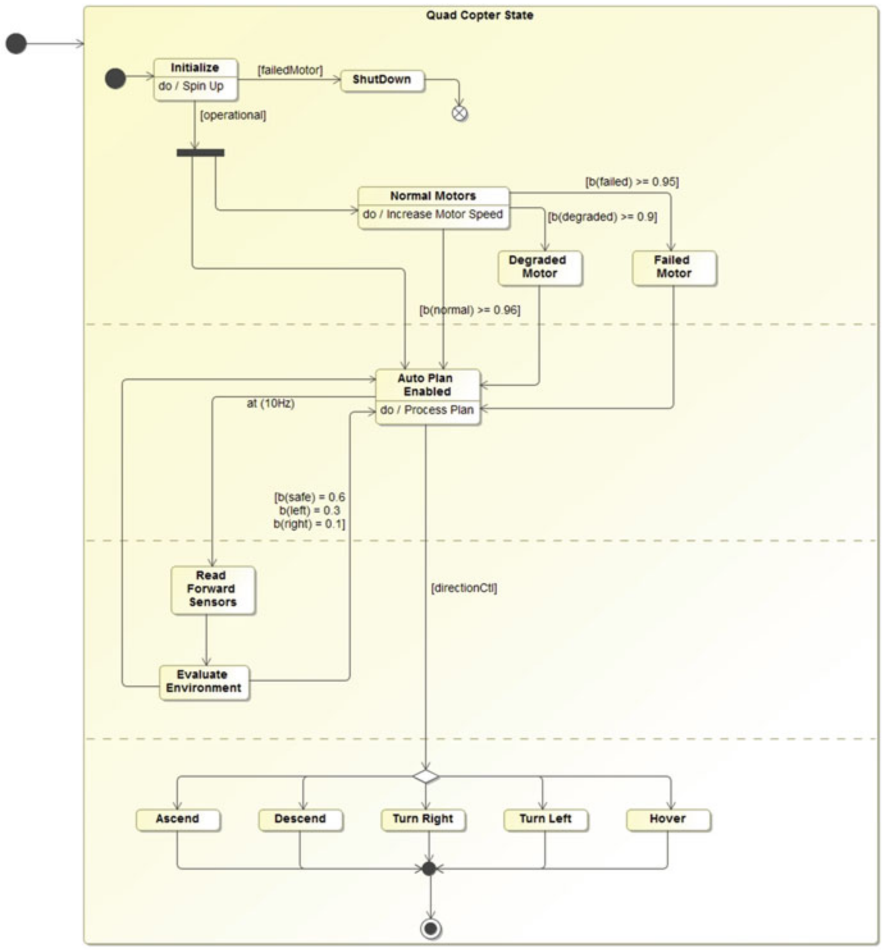


Fig. 29.4 State transition diagram for multi-UAV SoS

commanded trajectory that drive the position controller, an attitude controller that accepts inputs from the position controller and gyros. The gyros sense attitude (roll, pitch, and yaw) of the quadcopters and provide that information to attitude controller. The GPS senses the position and altitude of the quadcopter and provides that information to position controller.

The simulation of single UAV (quadcopter) and multi-UAV configuration is addressed in following steps: single quadcopter following a prescribed path; single quadcopter avoiding static obstacles to the left or right; single quadcopter avoiding a wall to the left; a single quadcopter avoiding a pillar to the left and a wall to the right (Fig. 29.8); multiquadcopter pursuing prescribed trajectories without obstacles; and multiquadcopters pursuing respective trajectories with obstacles (Fig. 29.9).

Table 29.1 UAV Swarm is a SoS

Operational independence of UAVs UAVs operate independently to satisfy mission requirements
Managerial independence of UAVs UAVs can be governed independently while being part of swarm
Evolutionary development of SoS Development and existence are evolutionary with functions and purposes added, removed, and modified with experience and need
Emergent SoS behavior UAV-SoS performs functions and carries out purposes that do not reside in any single UAV UAV-SoS behaviors are emergent – cannot be realized by a single AV
Geographic distribution UAVs are displaced in space and time and primarily exchange information

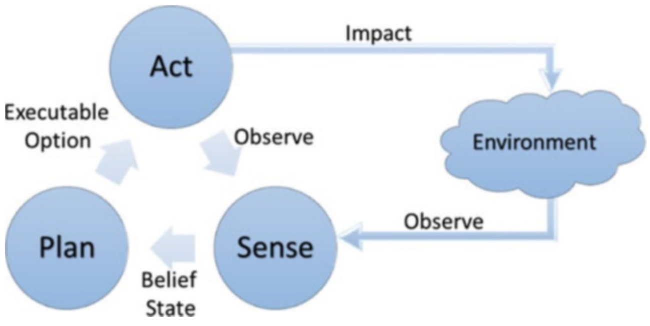


Fig. 29.5 "Sense-plan-act" modeling construct for UAV

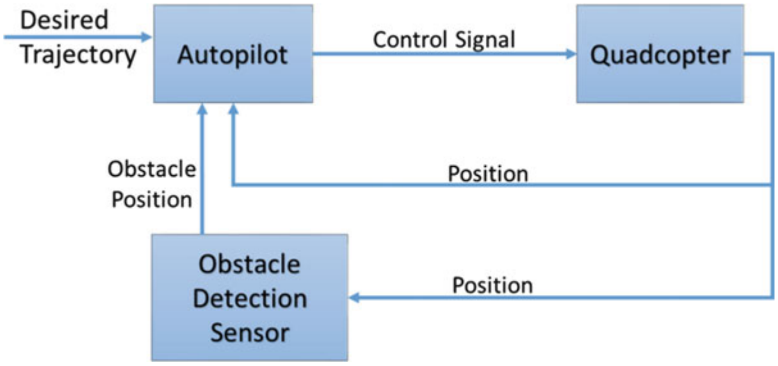


Fig. 29.6 Current implementation

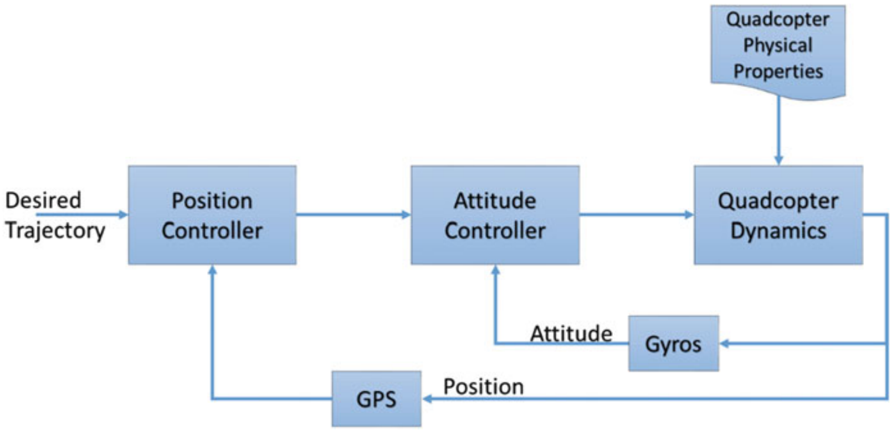


Fig. 29.7 Quadcopter functional architecture

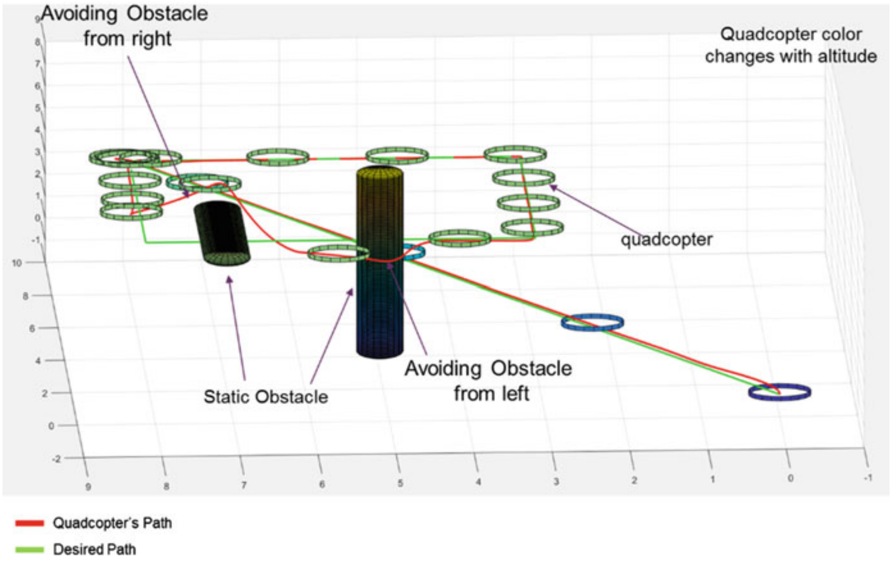


Fig. 29.8 Single quadcopter avoiding pillar from left and wall from right

29.7 Conclusions

This chapter has presented a resilient systems design approach based on a formal modeling approach. The approach combines CBD and POMDP to create a RC. The latter construct is well-suited to modeling complex systems in a scalable fashion with sufficient flexibility to incorporate resilience mechanisms. The approach supports verification of system and SoS models. The target application domain

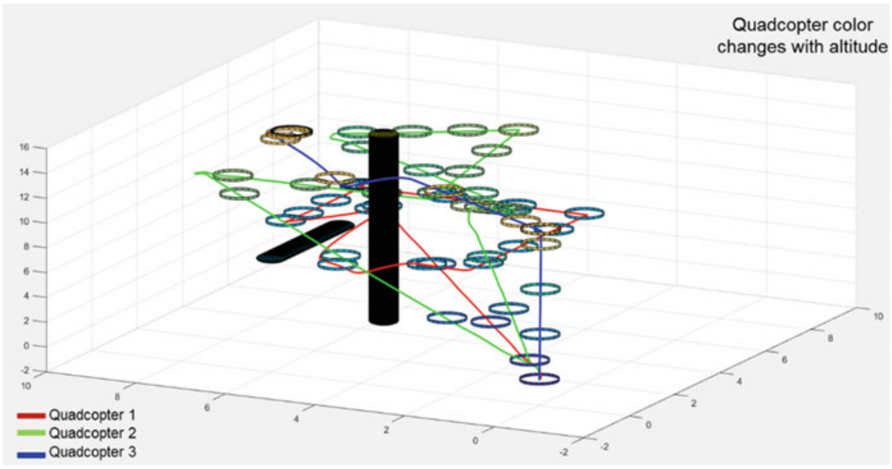


Fig. 29.9 Multiquadcopter following specified trajectories while avoiding obstacles

used to demonstrate RC is multi-UAV swarm control. However, the approach is sufficiently general to be applied to a variety of SoS including autonomous vehicle networks.

Acknowledgments This work was supported in part by Department of Defense, Systems Engineering Research Center, RT-166 contract no. HQ0034-13-D-0004.

References

1. Goerger SR, Madni AM, Eslinger OJ (2014) Engineered resilient systems: a DoD perspective. Procedia Comput Sci 28:865–872
2. Woods DD (2006) Essential characteristics of resilience. Resilience engineering: concepts and precepts, pp 21–34
3. Carlson JM, Doyle J (2000) Highly optimized tolerance: robustness and design in complex systems. Phys Rev Lett 84(11):2529
4. Csete ME, Doyle JC (2002) Reverse engineering of biological complexity. Science 295 (5560):1664–1669
5. Neches R, Madni AM (2012) Towards affordably adaptable and effective systems. Syst Eng 16 (2):224–234
6. Madni AM, Jackson S (2009) Towards a conceptual framework for resilience engineering. IEEE Syst J 3(2):181–191
7. Madni AM, Sievers M (2015) A flexible contract-based design framework for exaluating system resilience approaches and mechanisms. IIE annual conferenec and expo, Nashville, 30 May–2 June 2015
8. Sievers M, Madni AM (2014) A flexible contracts approach to system resiliency. Systems, Man and Cybernetics (SMC), IEEE international conference on 2014, IEEE

Chapter 30
Improving Lifecycle Product Data Management (LPDM) Within the US Army Research, Development, and Engineering Command (RDECOM)

Thomas W. Haduch, Robert S. Bruff, and Paul M. Martinell

Abstract This paper discusses the challenges of, and value in, implementing a lifecycle approach to management of the extensive and complex product data required in the engineering design, acquisition, and sustainment of military systems. The current state of Army management of product data and the future solution, the development of a well-integrated Lifecycle Product Data Management (LPDM) system, are discussed with three challenge examples and their related solutions (clean and accurate data, Configuration Management and control, and data sharing). The difference between a subordinate enterprise Product Data Management (ePDM) system, and an overarching LPDM system, is discussed, along with early designing for supportability (engineering) and the acquisition of supporting supplies and equipment (logistics). The paper is then summarized highlighting the advantage to the warfighter when a well-integrated LPDM is in place.

Keywords Lifecycle Product Data Management • Research, Development, and Engineering Command • Configuration Management • Enterprise Product Data Management

T.W. Haduch (✉)
RDECOM HQ, 3073 Aberdeen Boulevard, Room 105F, Aberdeen, MD 21005, USA
e-mail: thomas.w.haduch.civ@mail.mil

R.S. Bruff
e-mail: robert.s.bruff.ctr@mail.mil

P.M. Martinell
e-mail: paul.m.martinel.civ@mail.mil

A.M. Madni et al. (eds.), *Disciplinary Convergence in Systems Engineering Research*, https://doi.org/10.1007/978-3-319-62217-0_30

30.1 Background

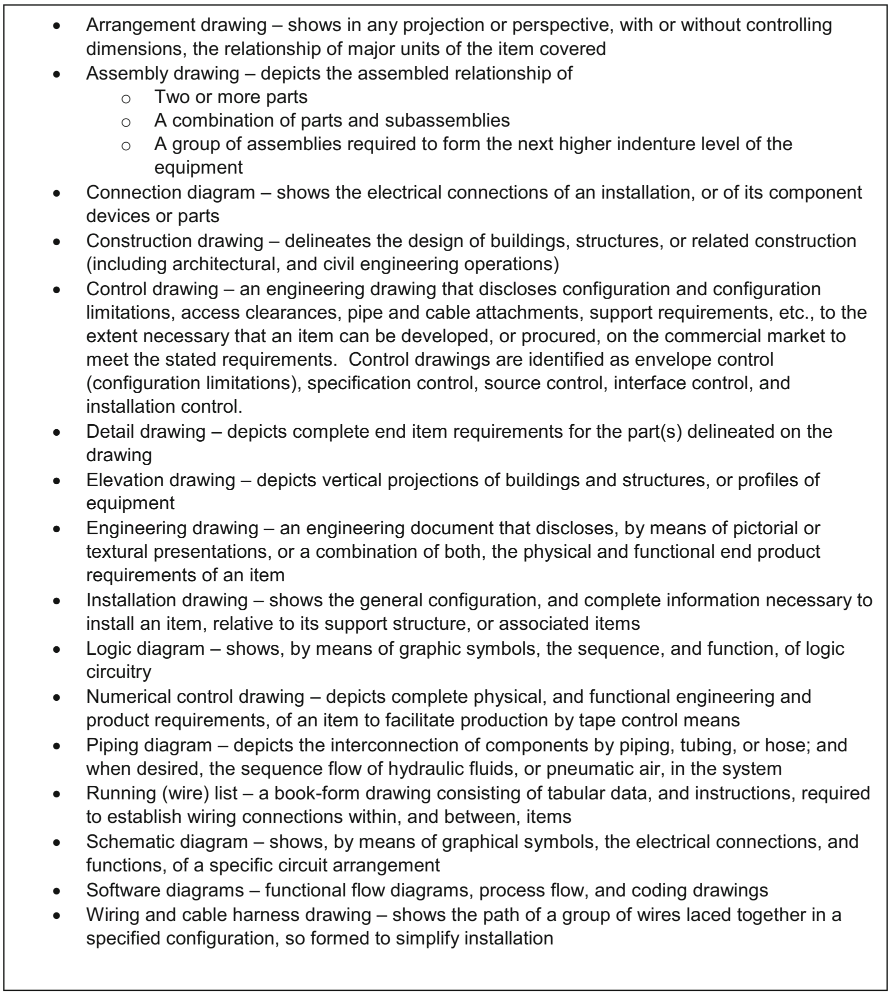
The methodology for documenting system design continues to change rapidly as a result of advances in information systems technology, and the digitization of previously exclusively hardcopy engineering documentation. These advances have increasingly promoted the use of electronic databases for the purposes of information processing, storage, and retrieval. Through the use of Computer Assisted Design (CAD) techniques, part, component, subsystem, and system designs and drawings can be stored in the form of three-Dimensional (3D) representations, two-Dimensional (2D) line drawings, in digital format, or a combination of any of these. By using computer graphics, word processing capability, email communications, cloud storage technology, and the like, designs can be presented faster, in more detail, and in easily modified formats.

Although many advances have been made in the application of computerized methods to data acquisition, storage, and retrieval, there continues to be lifecycle needs for integration of the information captured on design documentation. This information includes a combination of the following:

- Design drawings – assembly drawings, control drawings, logic diagrams, installation drawings, schematics, etc.
- Material and parts lists – part lists, material lists, long-lead-item lists, bulk-item lists, provisioning lists, etc.
- Analyses and reports – trade-off study reports supporting design decisions, reliability and maintainability analyses and predictions, human factors analyses, safety reports, supportability analyses, configuration identification reports, computer documentation, installation and assembly procedures, etc. [1].

Today, design drawings, constituting a primary source of system definition, may vary in form and function depending on the design objective, the method of information capture, and the source of the design information; that is, the type of equipment being developed, the extent of development required, whether the design is to be subcontracted, etc. Some typical types of drawings are specified in Fig. 30.1.

During the iterative process of detail design, engineering documentation is often initially rather preliminary, and then gradually progresses to the depth, and extent of definition, necessary to enable product manufacture, and full support over the total product lifecycle. Traditionally, the responsible designer, using appropriate design aids, produces a functional diagram of the overall system. The system functions are analyzed, and initial packaging concepts are developed. With the aid of specialists representing various disciplines (electrical, mechanical, components, reliability, and maintainability), and supplier data, detail design layouts are prepared for subsystems, units, assemblies, and subassemblies. The results are analyzed, and evaluated, in terms of functional capability, reliability, maintainability, human factors, safety, producibility, and other design parameters to assure compliance with the allocated requirements, and the initially established design



- Arrangement drawing – shows in any projection or perspective, with or without controlling dimensions, the relationship of major units of the item covered
- Assembly drawing – depicts the assembled relationship of
    - Two or more parts
    - A combination of parts and subassemblies
    - A group of assemblies required to form the next higher indenture level of the equipment
- Connection diagram – shows the electrical connections of an installation, or of its component devices or parts
- Construction drawing – delineates the design of buildings, structures, or related construction (including architectural, and civil engineering operations)
- Control drawing – an engineering drawing that discloses configuration and configuration limitations, access clearances, pipe and cable attachments, support requirements, etc., to the extent necessary that an item can be developed, or procured, on the commercial market to meet the stated requirements. Control drawings are identified as envelope control (configuration limitations), specification control, source control, interface control, and installation control.
- Detail drawing – depicts complete end item requirements for the part(s) delineated on the drawing
- Elevation drawing – depicts vertical projections of buildings and structures, or profiles of equipment
- Engineering drawing – an engineering document that discloses, by means of pictorial or textural presentations, or a combination of both, the physical and functional end product requirements of an item
- Installation drawing – shows the general configuration, and complete information necessary to install an item, relative to its support structure, or associated items
- Logic diagram – shows, by means of graphic symbols, the sequence, and function, of logic circuitry
- Numerical control drawing – depicts complete physical, and functional engineering and product requirements, of an item to facilitate production by tape control means
- Piping diagram – depicts the interconnection of components by piping, tubing, or hose; and when desired, the sequence flow of hydraulic fluids, or pneumatic air, in the system
- Running (wire) list – a book-form drawing consisting of tabular data, and instructions, required to establish wiring connections within, and between, items
- Schematic diagram – shows, by means of graphical symbols, the electrical connections, and functions, of a specific circuit arrangement
- Software diagrams – functional flow diagrams, process flow, and coding drawings
- Wiring and cable harness drawing – shows the path of a group of wires laced together in a specified configuration, so formed to simplify installation

Fig. 30.1 Typical engineering drawing classifications [2]

requirements. This classical Systems Engineering (SE) driven review and evaluation occurs at each stage in the basic design sequence, and generally follows the steps presented in Fig. 30.2 [2].

Engineering data are reviewed against standards and checklist criteria, throughout the industrial/government sectors, design standards, in the form of manuals and handbooks, are developed to cover preferred component parts and supplier data, preferred design and manufacturing practices, designated levels of quality for specified products, and requirements for safety. These standards, as applicable, may serve as a basis for design review evaluation [3]. This process addresses the engineering design of systems, but the ability of any design (and its attendant supporting documentation) to adequately logistically support the full lifecycle of

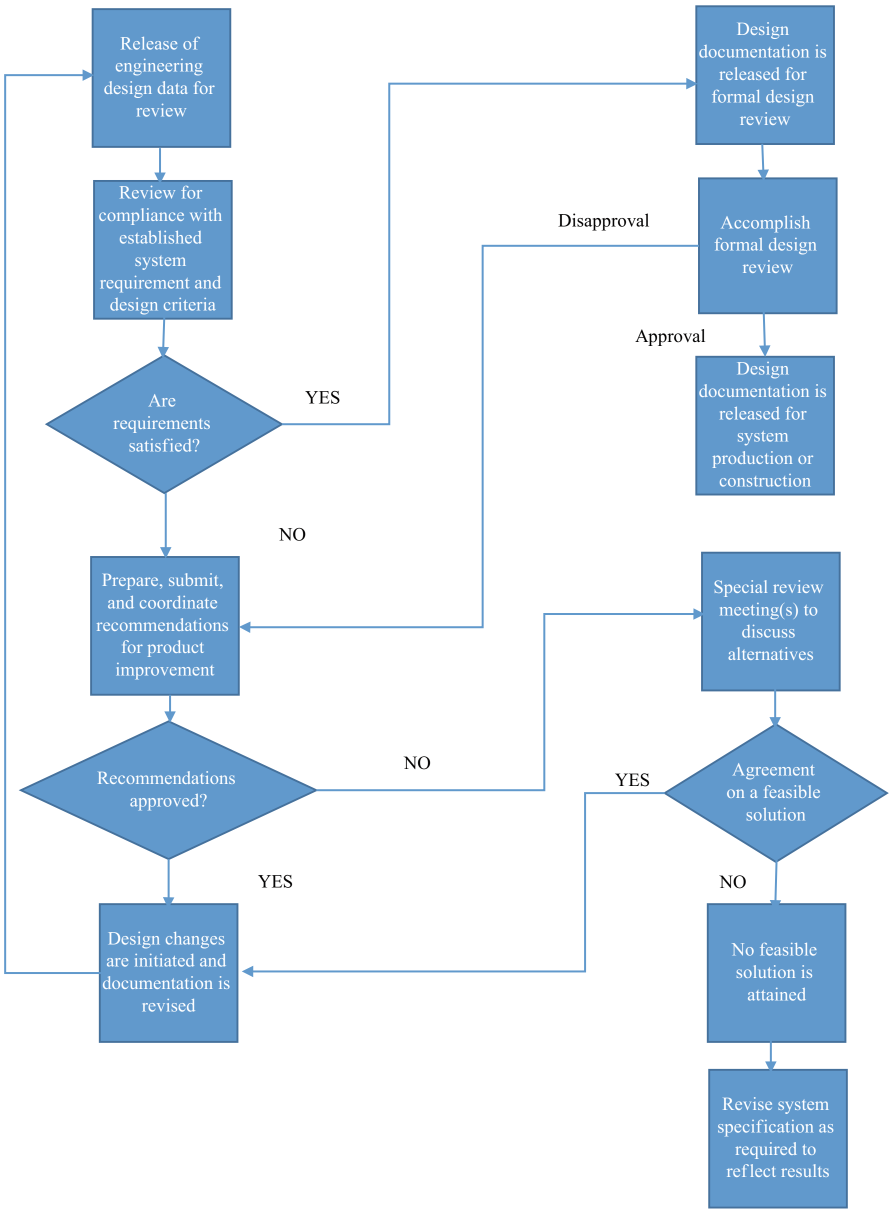

Release of engineering design data for review
Review for compliance with established system requirement and design criteria
Are requirements satisfied?
YES
NO
Design documentation is released for formal design review
Accomplish formal design review
Disapproval
Approval
Design documentation is released for system production or construction
Prepare, submit, and coordinate recommendations for product improvement
Recommendations approved?
NO
YES
Special review meeting(s) to discuss alternatives
Agreement on a feasible solution
YES
NO
Design changes are initiated and documentation is revised
No feasible solution is attained
Revise system specification as required to reflect results

Fig. 30.2 Design data review cycle

a system (cradle to grave (including disposal)) is too often lacking. The opportunity now presented, with most design documentation in digital format, is to seamlessly share this information, in an optimized and agile manner, throughout the system lifecycle. The challenge is that there are often legacy imbedded islands of this approach, using standalone data systems, which either support logistical downstream considerations in a cumbersome manner, or not at all.

30.2 The Current State: Challenge

Overview – The US Army lacks an integrated process for managing lifecycle weapon systems, and end item data across the enterprise. Data exist in multiple domains, across disparate networks and systems; and, in too many cases, in nonstandard formats. The deployment of Enterprise Resource Planning (ERP) systems, specifically the Logistics Modernization Program (LMP) and Global Combat Support System – Army (GCSS-Army) have served to highlight challenges for the Army with respect to Data Management (DM). The first challenge often begins when the Army does not acquire the "right" data, or includes standard contract clauses critical to insuring that the government maintains rights to the data required to support a system throughout its lifecycle. The data affected may include, but is not limited to, data related to Bills of Materials (BOMs) and components of end items, associated support items of equipment, and basic usage items. Disparate engineering data systems include nonintegrated legacy Product Data Management (PDM) systems, Interactive Configuration management and Procurement Program (ICAPP), Engineering Data Information Server (EDIS), and Multiuser Engineering Change Proposal.

Automated Review System (MEARS) and Logistics Information Systems (LISs) (Integrated Lead Time Management and Reporting System (ILTMARS), and Intelligent Interactive Logistics (I2LOG)).

30.2.1 Challenge Example 1

LMP and GCSS-Army require clean and accurate data to support development of BOMs, including the Provisioning Bill of Materials (PBOMs), Manufacturing Bill of Materials (MBOMs), and Repair Bill of Materials (RBOMs), as well as execution of sustainment logistics functions. Solving the problem requires elimination of the multiple, disparate DM methodologies across both the acquisition and logistics domains. To make this picture fully complete, similar consideration must be given to a system even as it is in the Research and Development (R&D) phase. Indeed, it is even more important that any system get off to a good start and consider the full logistic tail (including even ultimate disposal) as early in the process as possible. While engineering is organizationally and systemically separate from provisioning and maintenance (an understandable structural convenience), respective representations of the BOM are not synchronized, resulting in a lack of a valid configuration baseline for any downstream BOM (a luxury we can no longer afford, for either cost efficiency or system sustained agility). In addition, individual and separate maintenance of the BOM by engineering, provisioning, maintenance, and tactical units, results in duplicate processes, and compromises configuration control.

30.2.2 Challenge Example 2

One of the main issues affecting sustainment support today is Configuration Management (CM), and control. Specifically, in the Complex Assembly Manufacturing Solution (CAMS), the depots update "As Built," "As Received," and "As Maintained" BOMs after an item is manufactured, repaired, or rebuilt. This data is then passed to the LMP's Enterprise Central Component (ECC). The equipment master, within ECC, serves as the authoritative source of system configuration. However, today, there is a gap between the logistics enterprise (LPM and GCSS-Army) and the engineering centers in sharing data, which results in a loss of configuration control, and wasted man-hours.

30.2.3 Challenge Example 3

Army engineering centers independently develop and manage millions of 2D files, and 3D data objects, within unsynchronized legacy, and stove-piped PDM systems. These systems do not allow for data sharing using standard processes, and formats. One example is the inability of the Prototype Integration Facility (PIF) and Organic Industrial Base (OIB) to perform rapid prototyping via electronic exchange of data (email, Compact Disc (CD), file exchange, and SharePoint), and incurs the cost to share and manage that data outside of enterprise systems. This hampers the Army's ability to effectively, and efficiently, manage weapon systems throughout the lifecycle.

30.2.4 Challenge Summary

Program Executive Officers (PEOs)/Program Managers (PMs) are burdened by fragmented PDM systems, which inhibits collaboration and a well-integrated Systems of Systems (SoS) approach. Further, too often, the logistics community does not have consistent, and timely, access to authoritative data [4].

Today, product data for Army weapon systems are often managed in multiple disparate systems. Stakeholders exercising authority over systems, processes, and product data act independently, and in a nonintegrated way. Multiple Information Technology (IT) solutions (Commercial off-the-Shelf (COTS)/Government off-the-Shelf (GOTS), ERP) are in Army portfolios. Additionally, there is significant use of Original Equipment Manufacturer (OEM) PDM systems to provide support information.

This creates a multitude of nonstandard (and often ad hoc) processes, duplicate or missing product data, and degradation of CM; thus, increasing sustainment, and lifecycle, cost. Employed supplier networks and systems are not registered in the Army Portfolio Management System, and current engineering PDMs cannot fully support the logistics sustainment processes and ERP systems. These supplier systems may not have the required embedded security features typical of a system supporting military platforms. Both response time and cost go up; and the use of supplier data systems "locks" the military into a specific supplier for fear of losing the data. Finally, multiple stakeholders are "solving" the problem in disjointed, independent, nonintegrated (and often nonagile) ways.

30.3 The Future State: Solution

30.3.1 Vision

The Lifecycle Product Data Management (LPDM) vision is to have an internal, integrated capability to effectively manage Army weapon system product and end item data, throughout the lifecycle; thus, providing a cost, and time, responsive End-to-End (E2E) solution. This vision requires combining understanding the data sources (OEMs and RDECs) and business processes of the stakeholders and architecting the Information Technology (IT) infrastructure to support the LPDM system. The increase of collaboration across the lifecycle will be enhanced by the use of enforced standardized, or converted, common data formats and processes. Integrated BOMs will be established supporting the acquisition, and logistics, processes. Finally, the LPDM system will enable emerging Army information enterprise capabilities, and promote the concept of agility.

Shifting away from a linear, document-centric acquisition process toward a dynamic digital model-centric ecosystem provides direct lifecycle benefits to the warfighter. Digital models can easily encompass data, algorithms, process flows, or a hybrid blend. It allows a shift from low fidelity, implicit representations to one of high fidelity, explicit models, serving as the "single source of truth" for all stakeholders. Documents shift from the primary role of specification to the secondary role of communication. Lifecycle model-centric engineering (another way of looking at LPDM) allows a shift from today's stove-piped data sources to a future state of an agile, dynamic, and digital model-centric ecosystem [5].

The term digital thread, coined at Lockheed Martin, is the concept of an unbroken data link that stretches back to the original computer model of a part; the unbroken data path is the digital thread. A newer term, digital tapestry, also a term credited to Lockheed Martin, ties everything in a production operation together digitally, from concept to product realization. It is the E2E digital approach where everything is connected – from concept, design, simulation, manufacturing, and assembly, to testing and getting the final product to the customer [6].

Another documented success of this process is that of Microsoft's One PDM, in which their team demonstrated a reduced time to market, and increase agility within the marketplace [7]. Further, with decreasing military budgets for new hardware as the driving force, the seamless integration of maintenance and reliability data has informed such programs as the Navy's Service Life Extension Program (SLEP) with real-time field data [8]. All these programs operate under the concept the Army intends to implement with LPDM.

The LPDM Integrated Product Team (IPT) charter was developed based on agreement between the Army Materiel Command (AMC), Acting Executive Deputy to the Commanding General (EDCG), Department of Army (DA), Assistant Deputy Chief of Staff (ADCS) G-4, and Assistant Secretary of the Army (Acquisition, Logistics, and Technology) (ASA(ALT)), Principle Deputy; via email dated August 4, 2015. This charter established the LPDM IPT and was effective upon signature of the four IPT voting members. This LPDM IPT executed an initiative to implement LPDM capabilities supporting E2E Army business processes [9].

A subset of LPDM is enterprise Product Data Management (ePDM). This system is an Army initiative to create the infrastructure needed to manage all the information related to a product across the lifecycle, considering all stakeholders, less logistics (sustainability). The ePDM capability supports Science and Technology (S&T), Systems Engineering (SE), system development, and acquisition logistics business needs to include:

- Technology assessment/development
- Prototype integration
- Modeling and Simulation (M&S)
- Design
- CM
- Trade studies
- Logistics Support Analysis (SA) [10]

A few clarifications (definitions really) are in order here. Classical SE considers design for supportability in the field environment; this is, therefore, a part of ePDM. Logistics manages acquisition of replacement parts and materials that are required to execute supportability; this is, therefore, a part of the larger LPDM. Within ePDM, model-based SE uses M&S to enhance engineering analytical capabilities. It enables more effective and efficient systems development processes by specifying a system as a single evolving computer model, not a series of disconnected, static documents [11]. The synergistic effect of ePDM and Single Army Logistics Enterprise (SALE) may be visualized by considering Fig. 30.3.

Executing 74% of the Army's S&T budget, Research, Development, and Engineering Command (RDECOM) needs the enterprise capability to support not only current processes and operations but also desired future capabilities of:

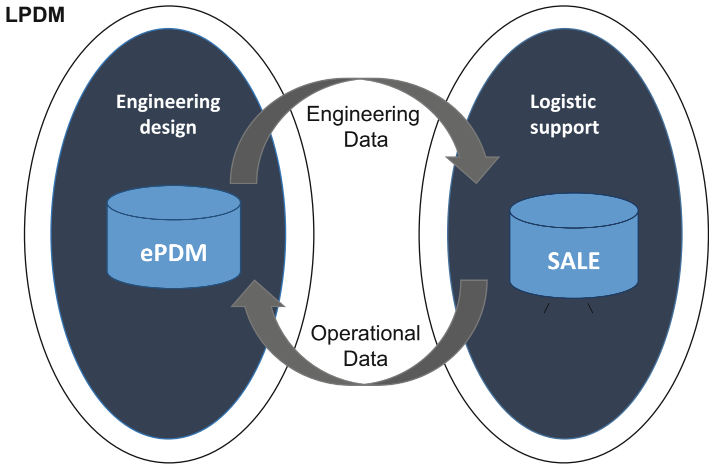


Fig. 30.3 The synergistic relationship of ePDM and SALE

- Design collaboration between and within Research, Development, and Engineering Centers (RDECs) and the Army Research Laboratory (ARL)
- Enhanced standardized product data management processes
- Enhanced reuse of engineering data (where appropriate)
- Enhanced support to downstream enterprise business processes in logistics and the organic industrial base [12]

Classical SE typically uses the simplified term of SA rather than Logistics SA [13]. This avoids the possibly confusing term, Logistics SA. In the correct context for this paper, a better, but also somewhat odd term, might be supportability SA. Either way, as used here when considering the ePDM context within the more encompassing LPDM context, SA constitutes the integration and application of different analytical techniques and methods to solve a wide variety of supportability problems. SA, in its application, is the process employed, on an iterative basis, through system development, that addresses the aspect of supportability in design. As such, it is an inherent part of the SE process and uses the results of reliability analysis (M&S; Failure Mode, Effects, and Criticality Analysis (FMECA)/Fault-Tree Analysis (FTA); and other standard predictions), maintainability analysis (Reliability-Centered Maintenance (RCM), level of repair analysis, Maintenance Task Analysis (MTA), and other predictions), and human factors analysis (Operator Task Analysis (OTA), Operational Sequence Diagrams (OSDs), error analysis, safety and hazard analysis, and training requirements) [14].

30.3.2 Solution Organizational Structure

Headquarters (HQ) Army Materiel Command (AMC), with representatives from the RDECOM; and Army G4 have formed an LPDM Integrated Product Team (IPT). RDECOM, with support from the Assistant Secretary of the Army (Acquisition, Logistics, and Technology) (ASA (ALT)), System of Systems Engineering and Integration (SoSE&I), submitted and received approval from the Office of Business Transformation (OBT) for the LPDM Problem Verification Form (PVF). The PVF approval authorizes preparation of the LPDM Problem Statement, Part 1. AMC drafted a charter to establish governance of LPDM through a Senior Executive Service (SES) level IPT, with representatives from the Department of the Army (DA) G4, HQ AMC, RDECOM, and ASA (ALT) Acquisition Policy and Logistics, as signatories. The charter was approved by all signatories on August 25, 2016.

30.3.3 Solution Example 1

Industry best practice recommends organizations achieve a single digital master BOM with multiple views for downstream use. Why should the Army settle for anything less in supporting the warfighter? The Army ERPs are close to achieving this end state, but are lacking the integrated engineering BOM, equipment master BOM, and key technology components. Enhancing Army ERPs as part of the LPDM initiative will resolve this capability gap by providing the tools, processes, and procedures required to manage Army weapon system and End Item (EI) data across the enterprise.

30.3.4 Solution Example 2

LPDM establishes a feedback loop to update EBOMs, and provide real time Engineering Change Management (ECM) and configuration controls by ensuring users are utilizing the latest technology package.

30.3.5 Solution Example 3

The LPDM initiative will provide an integrated system for CM of engineering data across the lifecycle of a weapon system as well as provide OIB and all ERP users an enterprise system for DM with a seamless access to standardized, shared, and secured data.

30.4 Additional Research and Tasking

The next step in this development process is to use a demonstration server to essentially “kick the tires” of the envisioned ePDM system. Such a demonstration server would verify that the ePDM system design does indeed fully and fluidly meet all stakeholder performance requirements and provide more data for continued research. This physical testing would employ user group approved methods and metrics to be developed. Testing must be designed to insure that the ePDM system meets stakeholder needs under real-life conditions, and over the entire lifecycle of the demonstration program(s).

This testing would establish a cross-functional team to identify common data and business process work flows and interface optimally between ePDM and SALE (the logistics element of LPDM). Further, this test/demonstration must align with ASA(ALT)’s suggestion of one overarching LPDM problem statement, and integrate well with in place elements of the full LPDM system. Finally, a LPDM Program Objective Memorandum must be established and approved to provide appropriate funding.

30.5 Summary

The engineering design of military equipment today is a complex undertaking involving a great deal of information from a large number of sources. Add to this the extended lifecycle of military equipment, and the need to provide for sustainability, often in hostile environments. As digitization has been applied to this effort over the last few decades, it has inevitably been with a focus on small pieces of the design and life cycle process. Lastly, with equipment supplied by vendors at all levels, budget constraints or nonholistic thinking has too often resulted in the acquisition of insufficient data packages and rights. When this shortcut was addressed, it was brushed off by the hope that required downstream data could be supplied by the vendor, when and if required. All this inconsistency, and somewhat shortsighted acquisition policies, led to the assortment of systems we have today, in which the required data are costly, fragmented, and its use does not lend itself to rapid response – either for design upgrades or logistical support.

The LPDM program described herein seeks to resolve those disparities by establishing a cradle to grave-integrated digital management framework for Army weapon systems. It allows an integrated lifecycle approach, and reduces overall cost, while enhancing the ability to respond quickly to unforeseen contingencies in a cost effective manner.

The solution set then is to (1) understand our data, including sources and formats (from OEMs and RDECs); (2) understand our business process; and (3) understand our IT infrastructure to fully support both. The ePDM program, a subset of LPDM, does this from the engineering point of view. Embedded within LPDM (the logistic piece) a total integrated, responsive lifecycle data management program is optimized.

References

1. Trade-offs and analyses are accomplished throughout the design process following classical Systems Engineering conceptual system design and preliminary system design. It is important that the results of these analyses be adequately documented to support design decisions. The reports identified here may be newly generated, or reports prepared during conceptual and preliminary system design that have been updated to reflect new information
2. Blanchard BS, Fabrycky WJ (1998) Systems engineering and analysis, 3rd edn. Prentice Hall, Upper Saddle River
3. Such standards may include ISO or ANSI standards or those prepared by professional societies (Institute of Electrical and Electronic Engineers (IEEE), American Society for Quality Control (ASQC), Society of Manufacturing Engineers (SME), American Society of Civil Engineers (ASCE)) or industry associations (Electronic Industries Association (EIA))
4. Haduch TW, Abaie M (2016) "Lifecycle Product Data Management (LPDM)," Presentation delivered to Christopher J. Lowman, Deputy Assistant Secretary Acquisition Policy and Logistics, 2 June 2016; and Thomas W. Haduch and Michael Abaie, "Lifecycle Product Management (LPDM)," Presentation delivered to Brigadier General (BG) Thomas H. Todd III, 28 June 2016
5. Baldwin KJ (2016) Systems Engineering Overview. Presentation delivered at the Conference on Systems Engineering Research (CSER), 22 March 2016
6. Randall S. (2016) Newton, "A digital stitch in time." *DE: Technology for Optimal Design Engineering*, July 2016
7. Stackpole B (2016) Microsoft flexes its Agile Muscles. *DE: Technology for Optimal Design Engineering*, July 2016
8. Taylor DP (2016) Keep them flying. *Seapower*, May 2016
9. Lifecycle Product Data Management (LPDM) Integrated Product Team (IPT) Charter
10. Haduch TW (2015) RDECOM charts the way ahead for systems engineering. *Army Technology Magazine*, May/June 2015
11. Haduch TW (2015) Model based systems engineering. *Army Technology Magazine*, May/June 2015
12. "The US Army Research, Development and Engineering Command," Command presentation dated 16 May 2016
13. The scope of activity is sometimes identified by the term Logistic Support Analysis (LSA), Maintenance Analysis (MA), Maintenance Engineering Analysis (MEA), Maintenance Level Analysis (MLA), or something equivalent. However, the objectives are the same
14. For defense systems, SA is covered in MIL-HDBK-502, "DOD Handbook-Acquisition Logistics," Department of Defense, Washington, DC, May 1997. The data requirements, evolving from SA (or Logistics Management Information), are specified in Government Electronic and Information Technology Association (GEIA)-STD-0007, Rev B, dated 1 May 2013 (this replaces MIL-PRF-49506. The emphasis here is on the SA process

Chapter 31
Verification and Validation of Behavior Models Using Lightweight Formal Methods

Kristin Giammarco and Kathleen Giles

Abstract The research described herein provides a method for exposing invalid behaviors in systems of systems (SoS) early in design, at the architecture level. The Monterey Phoenix (MP)-based method for conducting behavior model verification and validation (V&V) was developed after students ranging from high school to the graduate level began discovering unintended, invalid, and potentially high consequence behaviors permitted by their designs. These unspecified behaviors were consistent with known requirements, but violated stakeholder intent. Examples from four models from different domains and developed by different students are presented, then used as a basis for developing a structured set of behavior model V&V criteria that may be applied to any MP model. Finally, the criteria are put into the context of a systematic method that guides modelers in a thorough V&V of the behavior model. The ease with which unspecified and potentially invalid behaviors were exposed by students at various levels of education suggests that this lightweight formal method approach for behavior model V&V is user-friendly for application by practitioners who have basic skills in logic and logical thinking. Follow-on work will further test the method on other MP behavior modeling efforts, with an aim to improve and extend behavior model V&V criteria and the methods in which they are employed.

Keywords Formal methods • Monterey Phoenix • Verification • Validation • Behavior modeling • System of systems

Monterey Phoenix Nomenclature

A: B C;	Ordered sequence of events (A includes B followed by C)
A: (B | C);	Alternative events (A includes B or C)
A: [B];	Optional event (A includes B or no event at all)
A: (* B *);	Ordered sequence of zero or more occurrences of event B in A
A: (+ B +);	Ordered sequence of one or more occurrences of event B in A

K. Giammarco (✉) • K. Giles
Naval Postgraduate School, Monterey, CA, USA
e-mail: kmgiamma@nps.edu; kbgiles@nps.edu

A.M. Madni et al. (eds.), *Disciplinary Convergence in Systems Engineering Research*, https://doi.org/10.1007/978-3-319-62217-0_31

A: {B, C};	Unordered set of events B and C in A (B and C may happen concurrently)
A: {* B *};	Unordered set of zero or more occurrences of event B in A
A: {+ B +};	Unordered set of one or more occurrences of event B in A

31.1 Introduction

Consider the following scenarios and their potential consequences: An order processing system enters a waiting state after a transaction is canceled [1]. A first responder administers rescue medication to an unconscious patient, unaware that the medication was already administered [2]. The International Space Station is unaware of a hazardous condition within a supply spacecraft as that spacecraft approaches to dock [3]. An unmanned aerial vehicle (UAV) on a search and track mission reaches a return-to-base condition and then finds and begins to track a new target [4]. These scenarios were exposed by students with mixed levels of modeling experience using lightweight formal methods for behavior modeling. After providing a brief background on verification and validation (V&V) and lightweight formal methods, this paper describes how the aforementioned scenarios presented in simulation, how the invalid behaviors were purged, general behavior model V&V criteria that were constructed as a result of these modeling efforts, and a systematic Monterey Phoenix (MP)-based formal method for using those criteria. The paper concludes with a summary and future work.

31.2 Related Work

Verification and validation are distinct processes used for ensuring that a system meets its requirements and specifications and satisfies the user's need. System verification is "the confirmation, through the provision of objective evidence, that specified requirements have been fulfilled" [5]. Verification is performed throughout a system's life cycle and involves performing tests to ensure the system continues to meet the requirements and specifications as the system develops. During design and development, verification may be performed solely with modeling and simulation and then augmented with actual system testing as prototypes and production models are fabricated. System validation is the procedure used to ensure compliance of any system element with its intended purpose [5]. As with verification, these methods that include test, inspection, measurement, and analysis are employed throughout the system's life cycle, typically at the end of each project milestone. Final validation is conducted on the completely integrated system, as a final exam prior to fielding.

Modeling and simulation are frequently used to support V&V activities throughout a system's life cycle. Model V&V should be conducted for each use of the model, as the model may be valid for one set of conditions but invalid for another [6]. Model verification ensures the implementation of the conceptual model is logically sound and that the conceptual model is programmed accurately into the computer model [6]. The two recognized verification approaches are model checking and theorem proving [7]. Model checking involves building a system-representative, finite model, and checking that anticipated states or behaviors correspond to the model, while theorem-proving methods define the system and its preferred attributes as mathematical logic-based formulas. Model checking is automatic but can result in a state explosion problem, whereas theorem proving can handle infinite state spaces but requires human interaction and is more time-consuming [7]. Model validation confirms the model generates outputs that accurately reflect the model's purpose. Model V&V should be performed each time the model is modified. Model validation can be accomplished several different ways: independent V&V where a third party decides whether the model is valid (typically the most costly) and validation by the model development team using test data or by the user.

In computer science and software engineering, formal methods are mathematically grounded techniques including logic, semantics, and formal languages [7]. The degree of formality varies between direct, logical interpretations using proofs and theorems and less rigorous methods that employ discrete mathematics notations to develop specifications [8]). Rushby classifies four levels of formal methods ranging from no use to theorem proving and proof checking using support tools and a completely formal specification language [8]. Formal methods can also be categorized in terms of breadth of application, from a widespread across all stages of the life cycle or applied to certain components or phases of system development [8]. Others use the term lightweight to characterize an approach used to analyze part of the specification document without re-baselining the entire specification and the term heavyweight to describe a deeper, complete application of the methodology [9, 10].

Lightweight formal methods have been used for the past few decades as a means for detecting errors in the initial stages of software development [9, 11]. The methodical, mathematical techniques and abstraction, which are innate to formal modeling, enable complexity to be reduced. For these reasons, formal methods were first applied to requirements engineering where specifications were dominated by natural language, and few tools existed for efficient analysis. Easterbrook et al. describe three case studies where lightweight formal methods were applied to requirements modeling for fault protection software requirements on NASA systems [9]. In these case studies, lightweight formal methods were applied selectively to the most critical requirements, resulting in the discovery of errors not detected using traceability analysis or inspection. The deficiencies were then corrected during the development phase, when changes are more easily managed and less costly [9]. Other examples of lightweight formal methods in system requirements development include IBM's Customer Information Control System in the 1980s, a new display information system for the UK Civil Aviation Authority's air traffic management system, and a requirements specification for the Traffic Collision Avoidance System (TCAS II) developed by the Safety-Critical Systems Research Group [7].

Lightweight formal methods have also been used in software architecture design, where abstraction allows designers to develop a conceptual model that can be used to capture interactions between components before the component details have been specified [10]. The application of formal method techniques for software testing provides value by using formal assertions to verify desired outputs and thus supporting system verification. Previously, use case testing and prototyping were the primary test methods and could only partially cover possible software outputs [8].

Well-established formal methods tools such as the Vienna Development Method (VDM) [12], Larch [13], and the Z specification notation [14] specify sequential system behavior in terms of relations, sets, and functions. Likewise, Communicating Sequential Processes (CSP) [15], I/O Automata [16], and Temporal Logic [17] specify behavior of concurrent systems [7]. The FORMAN (FORMal ANnotation) approach uses the concepts of event grammar and event hierarchies to build system behavior models to formalize universal assertions for defining debugging rules [18]. More recently, the Monterey Phoenix provides a framework for business processes and software system architecture design based on behavior models [19, 20].

Current industry standards for modeling system behavior include the Systems Modeling Language (SysML) viewpoints for sequence, activity, use case, and state machine diagrams, system dynamics (SD) models depicting control and feedback in system processes, and agent-based models (ABM) that describe agent behaviors and interactions between agents and with the environment. MP augments a typical ABM approach by adding standardization for defining agents and events using formalized event grammar and structured syntax [20, 21]. SD models are mathematically rigorous, yet abstract enough for a wide variety of system applications, but they are best used for closed-loop systems in which component dependencies must be considered at the global level [22]. In contrast, MP separates the agent behaviors from their interactions, permitting the potential reuse of portions of models. A drawback of using sequence or activity diagrams is the difficulty in representing all possible behaviors in one diagram [23]. MP addresses this shortcoming by generating an output for each possibility, which is useful in V&V activities [20]. Woodcock et al. [10] propose that the use of formal methods looks to be increasing, but mainly confined to critical system development. MP provides an automated tool that can be used at various levels of abstractions, promoting a more widespread use of formal methods in system behavior V&V.

31.3 Example MP Models from Different Domains

The following models were developed by students ranging from high school to the graduate level. Each model is summarized in a standard format to facilitate interpretation. Due to its lightweight properties and ease of access, the Monterey Phoenix approach was used. In none of the cases were the students expecting to find the errors that they did during their own model V&V activity.

31.3.1 Pilcher's Order Processing System

Joanne Pilcher modeled, among other things, an order processing system as part of her Systems Engineering Management master's thesis on generation of the Department of Defense Architecture Framework (DoDAF) view from MP models [1]. The analysis objective of this MP model was to experiment with converting a state diagram into an MP behavior model. Pilcher was successful in demonstrating the representation of the OV-6b State Transition Description logic in her MP behavior model, by modeling both states and transitions as events with precedence relations inside one root event representing the order processing system itself. An interesting by-product of this experiment, however, was the exposure of an invalid behavior that was not present in the original model – a case where the order processing system returns to a waiting state after cancelation of an order. On the surface, this appears to be a simple modeling error, since such a transition is not present in the original model. It is tempting to make an immediate correction without any further discussion about it. If there were not an original, correct model to use as a guide, however, such an erroneous scenario would be a valuable find, since its presence means that the design as modeled permitted this unwanted state. The modeling error stimulates the discovery of an important explicit requirement for how the system should not behave. The absence of a transition in the state diagram conveys this requirement only implicitly. If a scenario ending in a "waiting" state were to occur in a "safety critical" or "cyber secure" system, this would be a very undesirable behavior indeed and worth writing an explicit requirement for preventing such behavior. Pilcher's MP model provides inspiration for the following requirement: "The Order Processing System shall end all started transactions in either the Cancelled or Delivered state" (Fig. 31.1).

The presence of such a wrong scenario in an early version of an MP model offers an explicit example of undesired system behavior with cost and vulnerability implications if it were allowed to occur in the actual system. Once Pilcher observed "that questionable result," she "identified design errors resulting in a revised design" [1] that matched the original state transition specification. Since the state transition error found in the MP model was not committed in the original model, the significance of its identification and correction may have been masked in a simple

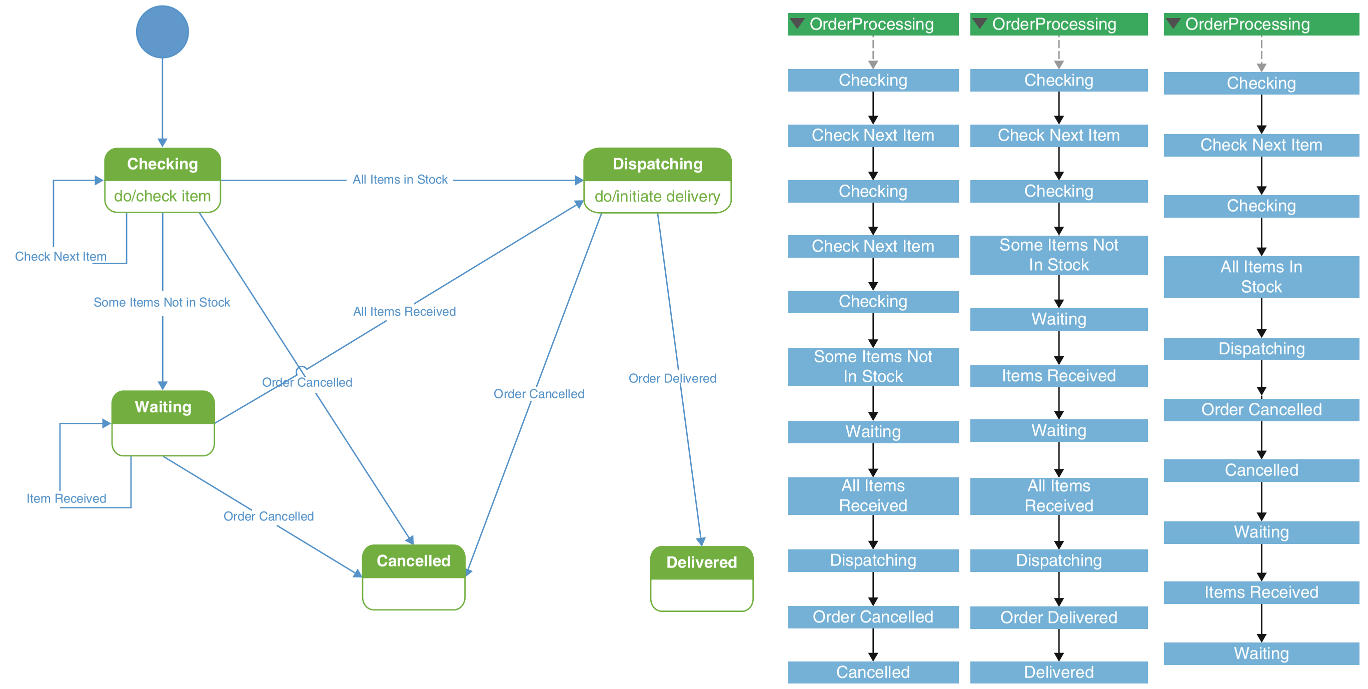


Fig. 31.1 *Left*, order processing state diagram (after Fowler and Scott [24]), and *right*, two example valid event traces (ending in *canceled* or *delivered*) and an example invalid trace (ending in *waiting*) discovered using the MP analyzer [1]

model verification activity, had the modeler not considered its presence as a possible violation of a significant unspecified valid behavior (a model validation activity).

31.3.2 Bryant's First Responder Process

Jordan Bryant, for her high school capstone research project, studied safety issues pertaining to a proposed process for layperson administration of a rescue medication called Narcan vis-à-vis existing regulations governing first responder behaviors [2]. Bryant's MP model produced various possible scenarios that could emerge based on the possible actions of bystanders and first responders and their interactions with the victim. The model included behaviors for the victim, the bystander, and a first responder and contained (among other variants) scenarios where a bystander administers the drug before arrival of first responders at the scene. The original analysis goal was to determine the time savings in having bystanders prepared to administer the rescue medication, but an unexpected behavior cropped up in simulation that neither student nor advisor considered. Among the automatically generated scenarios was a scenario in which the bystander calls 911 and then administers the Narcan, and then the first responder arrives and administers Narcan, apparently unaware that Narcan was already administered. This scenario prompted the idea to modify the proposed process to include the bystander marking the victim (e.g., by placing a medical bracelet that could be included with the kit) to indicate a dose and time administered visible to the first responders, which would reduce the risk of the victim receiving more Narcan than is necessary in the event the bystander is not present or involved in the situation when the first responder arrives. Bryant's MP model provides inspiration for the following requirement: "Any Bystander who administers Narcan to an Overdose Victim shall place a band around the Overdose Victim's wrist that indicates the amount and time of the Narcan dose administered" (Fig. 31.2).

For this modeling effort, there was no previous documentation to convert into an MP model – the MP model was the first formal attempt to document this process. There are also some verification-related errors in both of these traces, in the form of missing interactions. The validation-related error of the "administer Narcan" event occurring twice, once by the bystander and again by the first responder, stands out as a significant and potentially unsafe scenario. The presence of this scenario among other valid scenarios demonstrates, on a small example, the ability of MP to permute through all the combinations of actor behaviors far more comprehensively than a human would be able to do alone. Bryant concludes: "By inspecting the MP event traces, unexpected events, including miscommunication and patient response, can be identified and procedures refined before errors occur in real life" and "MP's use in predicting overdose scenarios could be beneficial to determining advantages to the medical field" [2].

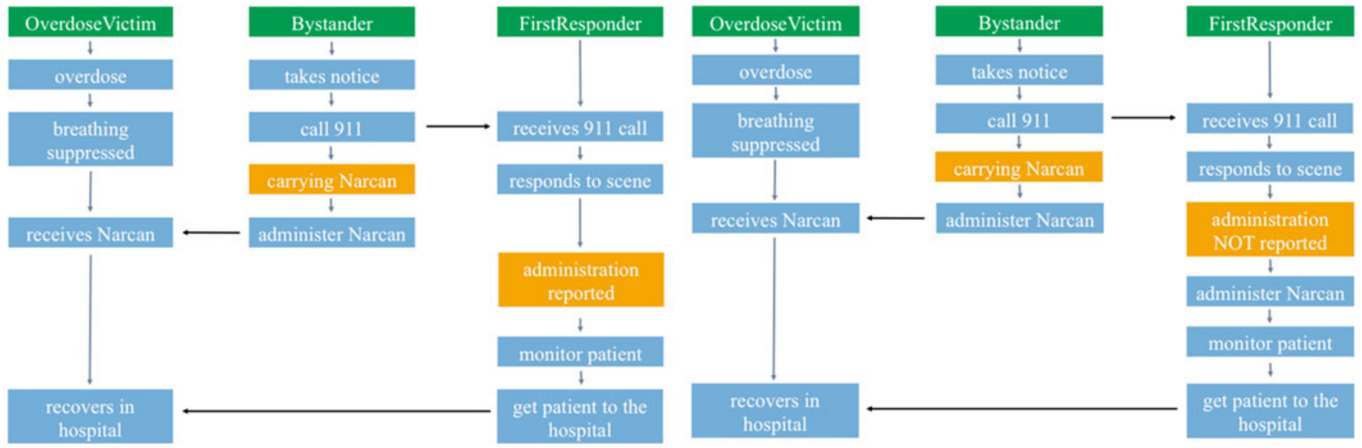


Fig. 31.2 *Left*, a "normal" scenario where all actors are behaving as expected according to the proposed process, and *right*, an unexpected scenario in which Narcan is administered twice (once by the bystander and again by the first responder), discovered using the MP analyzer [2]

31.3.3 Nelson's Spacecraft Communication System

As part of her master's project [3], Cassie Nelson modeled a spacecraft communication system that will be compatible with the International Space Station (ISS) for approach operations. The MP modeling effort was based on existing high-level system requirements for the spacecraft communication system upgrade. After launch, a resupply spacecraft approaches the ISS and docks. A successful rendezvous between spacecraft and ISS requires continuous and reliable communications throughout approach and docking, so Nelson's MP model focused on the communication aspect of the approach. A loss in communications could result in a high consequence failure, such as collision. The communication link status is monitored through a part of the transferred data packet called the heartbeat. Initially, Nelson found three valid scenarios: both spacecraft and ISS register a valid heartbeat and the approach continues, the spacecraft registers a valid heartbeat but the ISS registers an invalid one and both switch to a redundant communication system, and both spacecraft and ISS register an invalid heartbeat in which case the operation is aborted. Upon inspecting these three valid scenarios generated, it became clear to Nelson that a fourth scenario was missing from the specification: the one in which the spacecraft registers an invalid heartbeat and the ISS registers a valid heartbeat, in which case, the operation should be aborted. Nelson's MP model provides inspiration for the following requirement: "The ISS shall abort docking operations with a spacecraft that has an invalid heartbeat comparison, even if the ISS heartbeat comparison is valid" (Fig. 31.3).

This particular example demonstrates how MP may be used to discover errors of omission in requirements specifications. It takes a human analyst working in concert with an automated tool to uncover undesired and missing behaviors, the human doing inspection and validation tasks, and the automated tool presenting all combinations of possible behaviors based on the specification. Nelson concludes, "MP has the potential to impact systems engineering practices for complex systems. It is an aid to a systems analyst, provoking thoughts that they may not have otherwise had without a full scope set of event traces. The MP model in this study exposed unwanted behaviors due to missing system requirements. The ISS was expecting the spacecraft to continue approach because of the valid ISS heartbeat. However the spacecraft had an invalid heartbeat and therefore, the spacecraft began to abort operations" [3].

31.3.4 Revill's Unmanned Aerial Vehicle (UAV) System

Brant Revill, in his master's thesis [4], modeled a UAV system on a search and track mission as part of a larger effort to identify failure modes and fail-safe behaviors. An UAV may be part of a larger swarm that is launched to search and track objects of interest in the environment. One swarm operator controls all UAVs

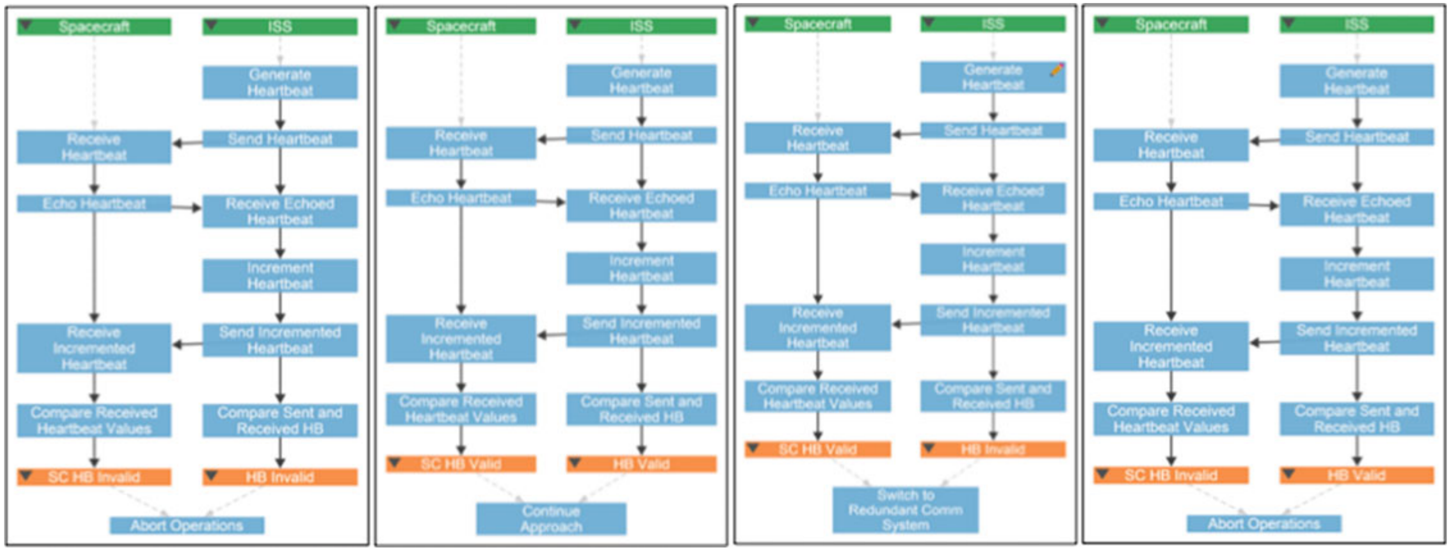


Fig. 31.3 All four scenarios are valid, but the one on the far *right* was not present until the modeler discovered its absence and modified the model to include it. The omission of this scenario would have resulted in an incomplete specification (what to do if the communication heartbeat registers as invalid on the spacecraft, but valid on the ISS) [3]

that are on the search and track mission, so the UAVs are required to have many autonomous behaviors to minimize the burden on the human operator. An example of a valid scenario is where the swarm operator commands an UAV to commence a search and track mission, an UAV detects an object in the environment, evaluates it, determines that it is the target of interest, and begins to track it. The UAV notifies the swarm operator, who then conducts an independent assessment of the validity of the target. The target turns out to be a valid target of interest in some cases and, in other cases, not a valid target (false alarm). The bingo fuel condition was also modeled, in which case an UAV has just enough fuel remaining to safely return to the landing site. Among the many scenarios, Revill's MP model produced one particularly questionable scenario, in which an UAV reached the bingo fuel condition and began its egress to the landing site and then it spotted a target and began to track it. This behavior may be valid, or it may be undesired, depending on how disposable the UAV is. Several possible requirements may be conceived as a result of this finding, such as "a UAV that has reached a bingo fuel condition shall request permission from the Swarm Operator to track any new targets found," "a UAV that has found a possible target after reaching bingo fuel shall relay the last known location of the target to the Swarm Operator, then continue to return to base," and "a UAV shall only track targets found before reaching bingo fuel conditions" (Fig. 31.4).

In any case, the discovery of this possibility in advance of experiencing the situation in actuality affords the designers some time to decide how the UAV should behave, if a target were to be presented after bingo fuel conditions are reached. Of further interest in the scenario depicted below, the tracked target was not even a valid target (as determined by the swarm operator), so target detection accuracy and UAV expendability should be related considerations. If target detection accuracy and vehicle expendability are both low, for example, this may constitute a high-risk scenario.

31.4 Behavior Model V&V Criteria

Table 31.1 summarizes verification and validation activities that can be conducted using Monterey Phoenix models. The first four verification activities were not explicitly discussed in the preceding examples, but were done as prework to producing traces for inspection. For example, syntax errors and typos are corrected after either direct inspection of the MP model or running the model and spotting them in the generated traces. Common example syntax errors are a missing semicolon or a misplaced parenthesis in the MP code. Common typos include misspelled event names and forgotten underscores in event names resulting in two separate events rather than a single, multiword event. Other verification activities include ensuring the absence of deviations from notation or style guidance, such as adhering to a particular language or naming convention. Example conventions may be to capitalize the first letter of events that are states, ensure

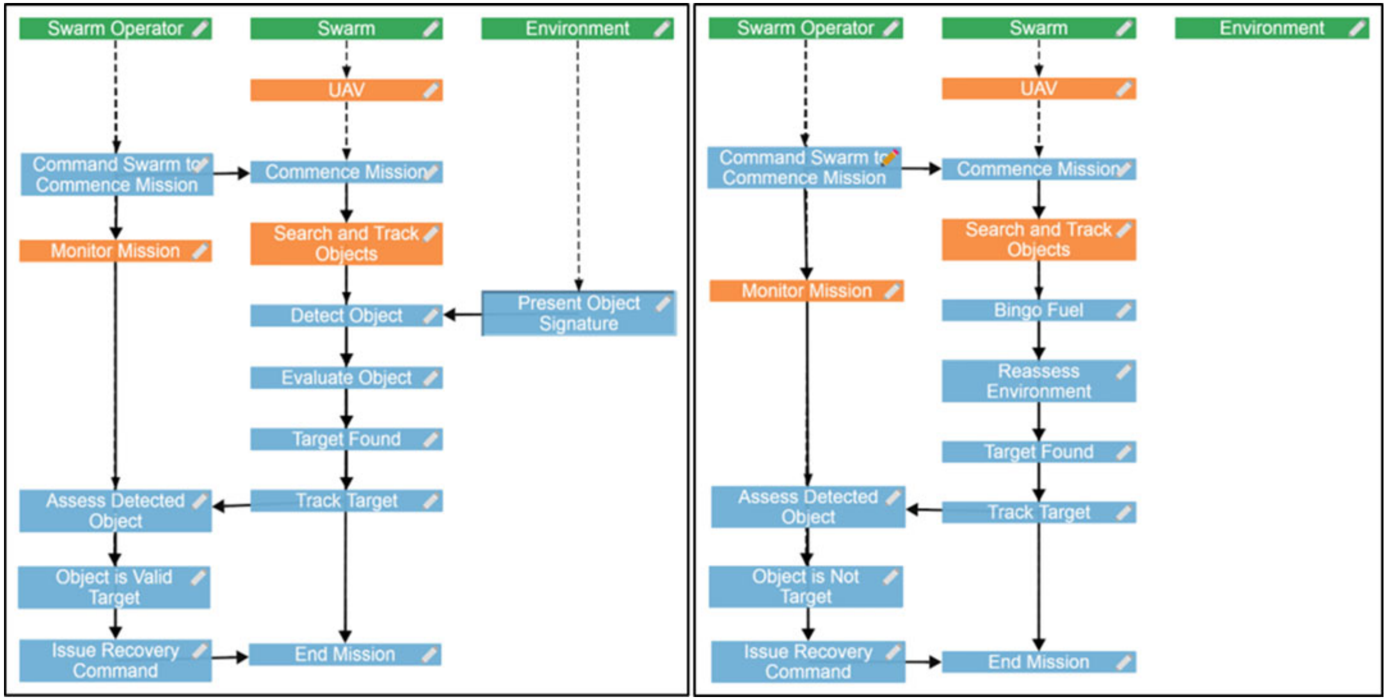


Fig. 31.4 *Left*, a “normal” scenario where all actors are behaving as expected according to the operational process, and *right*, an unexpected scenario in which a target is found and tracked after bingo fuel conditions are reached [4]

Table 31.1 Behavior model V&V criteria.

	Verification	Validation
Absence of	Syntax errors	
Absence of	Typographical errors	
Absence of	Deviations from notation and style guides	
Presence of	Correct scope size for the analysis	
Presence of	Specified valid behaviors	Unspecified valid behaviors
Absence of	Specified invalid behaviors	Unspecified invalid behaviors

states are grammatically phrased as continuous processes, lowercase all letters of events that are activities, ensure activities are grammatically phrased as active verbs, and ensure events that are actors are grammatically phrased as nouns. Conventions will vary based on architect preference but should be established and checked for consistency as part of the verification process so that grammatically confusing names do not mask underlying issues that would be otherwise exposed during validation. The fourth criterion in the table is to check for the presence of the correct scope size for the analysis. This is derived from experience with MP modeling; often, the modeled behavior looks fine at scope 1 (up to one iteration of loops in the model), but unwanted behaviors begin to appear at scopes 2 and higher. Jackson's small scope hypothesis that most errors can be found on small examples (or small number of loop iterations, in the case of MP) is leveraged here to expose many errors at a small scope cheaper and quicker than heavier methods like theorem proving [20].

The final four criteria in the table involve both verification and validation, and check for the presence of valid (wanted) behaviors, either specified (demonstrated through verification) or unspecified (discovered during validation). Specified valid behaviors are easy to relate to: stating the valid, desired behaviors is the predominant perspective taken in requirements specifications today. More difficult to relate to is the idea of finding unspecified valid behaviors permitted by the design – behaviors that were not explicitly required in the specification, but are nonetheless valid, and desired (such as in Nelson's ISS example [3]). With MP, we can also check for the absence of specified invalid behaviors, which can be done using assertion checking, where the requirement is formally posed as a statement and refuted using counterexamples from the set of scenarios where the requirement does not hold. The small scope hypothesis bounds this search space for a reasonable computation time, leveraging this feature of lightweight formal methods. Finally, we check for the absence of unspecified invalid behaviors, which typically has been the most elusive type of behavior to detect early in a system's design. What used to be like looking for a black cat in a dark room is now possible through manual or semiautomated inspection of generated scenarios, as illustrated in the earlier examples from Pilcher [1], Bryant [2], and Revill [4].

31.5 MP-Based Formal Method for Conducting Behavior Model V&V

From these early experiments that, quite accidentally, uncovered unspecified valid behaviors and unspecified invalid behaviors emerge a new formal method for conducting V&V on behavior models. The focus is on early discovery of unspecified behavior, so that we can take appropriate action on the unspecified behaviors we are seeing, whether that is explicitly specifying wanted behaviors (rather than risking assuming they will occur) or purging the unwanted behaviors (by adding constraints to the specification to serve as new requirements that inhibit unwanted behavior). Putting this V&V activity into a structured and repeatable process so that others can repeat these results on their own system models is the goal of this section, albeit briefly, for space constraints.

Steps of the method are as follows (illustrated in Fig. 31.5):

- Step 1: Gather documents and other available inputs, such as existing models, describing stakeholder requirements for system behaviors.
- Step 2: Describe the required behaviors in a Monterey Phoenix model.
- Step 3: Check the model against the verification criteria.
- Step 4: If verification issues are found in the behavior model, correct the discovered errors.
- Step 5: Run the model to generate the exhaustive set of scenarios and inspect manually or use automated tools such as assertion checking.
- Step 6: If a verification issue is found, correct the discovered error so that the model matches the intended specification. If a validation issue is found, review the discovered behavior with the appropriate stakeholder(s) and then make any necessary corrections. Repeat steps 5 and 6 as needed until all behaviors exhibited in all scenarios are accepted as valid.

The method is simple to represent in MP code, as shown in Appendix A. Additional background and description of the Monterey Phoenix approach and language may be found in prior publications [19, 20].

31.6 Conclusions and Way Ahead

This paper provided four example instances of discovery of unwanted behaviors using the Monterey Phoenix approach, and introduces a method for exposing invalid behaviors in systems of systems (SoS) early in design, at the architecture level. The ease with which unspecified and potentially invalid behaviors that were exposed by students at various levels of education suggests that this lightweight formal method approach for behavior model V&V is a user-friendly application for practitioners who have basic skills in logic and logical thinking. Follow-on work

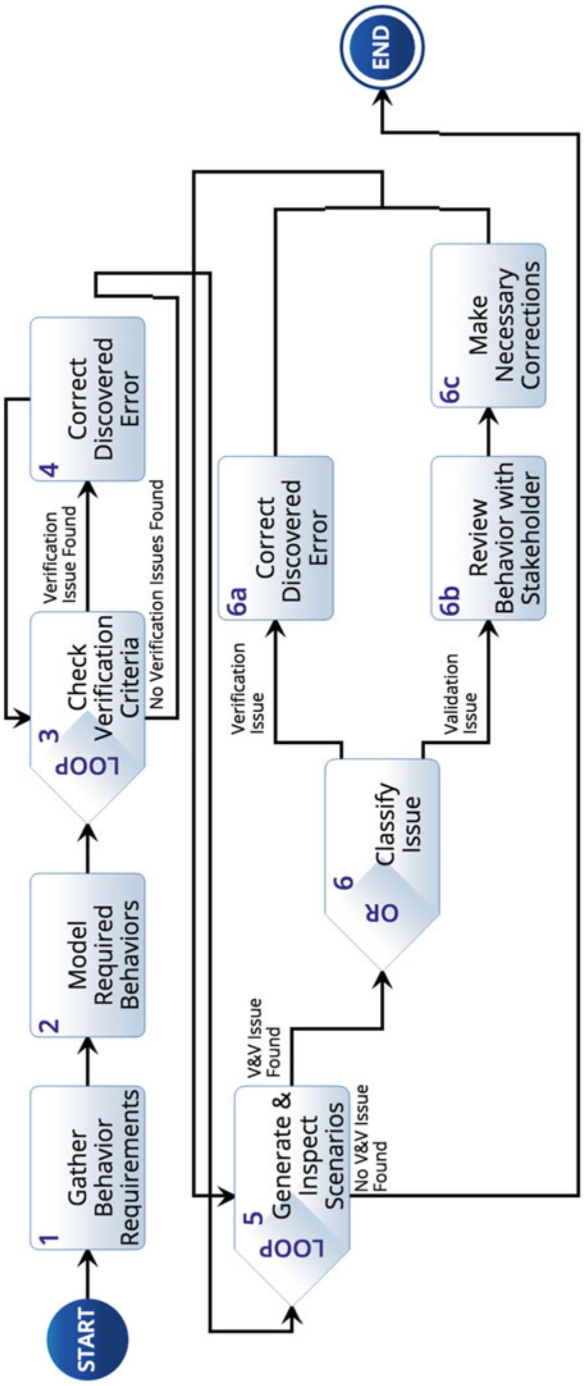

START
1 Gather Behavior Requirements
2 Model Required Behaviors
3 LOOP Check Verification Criteria
Verification Issue Found
4 Correct Discovered Error
No Verification Issues Found
5 LOOP Generate & Inspect Scenarios
V&V Issue Found
No V&V Issue Found
6 OR Classify Issue
Verification Issue
Validation Issue
6a Correct Discovered Error
6b Review Behavior with Stakeholder
6c Make Necessary Corrections
END

Fig. 31.5 A method for conducting behavior model V&V using the Monterey Phoenix lightweight formal approach and language

will further test the method on other MP behavior modeling efforts, with an aim to improve and extend the behavior model V&V criteria and the methods in which they are employed.

Acknowledgments The Consortium for Robotics and Unmanned Systems Education and Research (CRUSER) sponsored the development of the MP analyzer tool that enabled all of the student models and their discoveries.

Appendix A

Method for Conducting V&V with MP, Modeled Using MP

This simple MP model has one root and no interactions with other roots. This model generates six traces at scope one, 21 traces at scope two, and 60 traces at scope three on firebird.nps.edu.

```
SCHEMA        VV_Method_for_Behavior_Modeling
ROOT Modeler:   Gather_behavior_requirements     Model_required_behaviors
                Check_verification_criteria
                (* Verification_Issue_Found       Correct_discovered_error *)
                No_Verification_Issues_Found      Generate_and_inspect_scenarios
                (* VV_Issue_Found Classify_issue
                    ( Verification_Issue          Correct_discovered_error |
                      Validation_Issue          Review_behavior_with_stakeholder
                      Make_necessary_corrections )      *)
                No_VV_Issue_Found                 End;
```

References

1. Pilcher JD (2015) Generation of department of defense architecture framework (DODAF) models using the Monterey Phoenix behavior modeling approach. Master's Thesis, Naval Postgraduate School, September 2015
2. Bryant J (2016) Using Monterey Phoenix to analyze an alternative process for administering Naloxone. Capstone Research Project, Science and Math Academy, Aberdeen, MD. June 2016. Retrieved September 17, 2016 from http://www.scienceandmathacademy.com/academics/srt4/student_work/2016/bryant_jordan.pdf
3. Nelson C (2015) Modeling a spacecraft communication system using Monterey Phoenix: a systems engineering case study. Master's Project, Stevens Institute of Technology, Hoboken. November 2015
4. Revill MB (2016) UAV swarm behavior modeling for early exposure of failure modes. Master's Thesis, NPS, September 2016

5. SEBoK Authors (2016) System Verification. In BKCASE Editorial Board. 2016. *The Guide to the Systems Engineering Body of Knowledge (SEBoK)*, v. 1.6. R.D. Adcock (EIC). Hoboken: The Trustees of the Stevens Institute of Technology. *Released 23 March 2016, http://sebokwiki.org/w/index.php?title=System_Verification&oldid=50858 (accessed 18 Sep 2016)*
6. Sargent RG (2015) A tutorial on verification and validation of simulation models. In Proceedings of the 2015 Winter Simulation Conference, 1729–1740. http://dl.acm.org/citation.cfm?id=809441
7. Clarke EM, Wing JM, Alur R, Cleaveland R, Dill D, Emerson A, Garland S, Others (1996) Formal methods: state of the art and future directions. ACM Comput Surv 28(4):626–643
8. Rushby J (1993) Formal methods and the certification of critical systems. SRI International/ Computer Science Laboratory, Menlo Park
9. Easterbrook S, Lutz R, Covington R, Kelly J, Ampo Y, Hamilton D (1998) Experiences using lightweight formal methods for requirements modeling. IEEE Trans Softw Eng 24(1):4–14
10. Woodcock J, Larsen PG, Bicarregui J, Fitzgerald J (2009) Formal methods: practice and experience. ACM Computer Surveys 41(4):1–40
11. Agerholm S, Larsen PG (1999) A lightweight approach to formal methods. Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) 1641(October 1998):168–183. doi:10.1007/3-540-48257-1_10
12. Jones CB (1990) Systematic software using VDM. Prentice Hall, New York/London
13. Guttag JV, Horning JJ (2012) Larch: languages and tools for formal specification. Springer Science & Business Media, Berlin
14. Spivey, JM (1988) Understanding Z: a specification language and its formal semantics, vol 3, Cambridge University Press, Cambridge
15. Hoare CAR (1985) Communicating sequential processes, vol 178. Englewood Cliffs, Prentice-hall
16. Lynch NA, Tuttle MR (1988) An introduction to input/output automata. Massachusetts Institute of Technology, Cambridge, MA
17. Manna Z, Pnueli A (2012) The temporal logic of reactive and concurrent systems: specification. Springer Science & Business Media, Berlin
18. Auguston M. (1995) Program behavior model based on event grammar and its application for debugging automation. *AADEBUG*
19. Auguston M (2009) Monterey phoenix, or how to make software architecture executable. In Proceedings of the 24th ACM SIGPLAN conference companion on Object oriented programming systems languages and applications. ACM
20. Auguston M (2016) System and software architecture and workflow modeling language manual. Naval Postgraduate School, Monterey
21. Ruppel S (2016) System behavior models: a survey of approaches. Naval Postgraduate School, Monterey
22. Borshchev, A, Filippov A (2004) From system dynamics and discrete event to practical agent based modeling: reasons, techniques, tools. In The 22nd International Conference of the System Dynamics Society, vol 66, p 25–29
23. Auguston M, Giammarco K, Clifton Baldwin W, Crump J, Farah-Stapleton M (2015). Modeling and verifying business processes with Monterey phoenix. In 2015 Conference on Systems Engineering Research, vol 44, p 345–353
24. Fowler M, Scott K (1997) In: Carter Shanklin J (ed) UML distilled. Addison Wesley Longman, Reading, MA

Chapter 32
Categorical Foundations for System Engineering

Spencer Breiner, Eswaran Subrahmanian, and Albert Jones

Abstract In this paper, we argue that category theory (CT), the mathematical theory of abstract processes, could provide a concrete formal foundation for the study and practice of systems engineering. To provide some evidence for this claim, we trace the classic V-model of systems engineering, stopping along the way to (a) introduce elements of CT and (b) show how these might apply in a variety of systems engineering contexts.

Keywords Category theory • Foundations of system engineering • Mathematical modeling

32.1 Introduction

Systems are becoming more complex, both larger and more interconnected. As computation and communication in system components goes from novelty to the norm, this only becomes more true. In particular, we have no generally accepted method for designing, testing, and analyzing systems which mix both physical and computational dynamics. We believe that a new formal foundation is required to model and study such complex systems.

Existing approaches, typified by the V-model of systems engineering, are more heuristic than formal. First, we conceptualize the system, setting our various requirements and assumptions. Next, we refine this into a functional decomposition which details how our system will meet its goals. In realization, we map these

S. Breiner (✉)
National Institute of Standards and Technology, Gaithersburg, MD, USA
e-mail: spencer.breiner@nist.gov

E. Subrahmanian
National Institute of Standards and Technology, Gaithersburg, MD, USA

Carnegie Mellon University, Pittsburgh, PA, USA
e-mail: sub@cmu.edu

A. Jones
Carnegie Mellon University, Pittsburgh, PA, USA

A.M. Madni et al. (eds.), *Disciplinary Convergence in Systems Engineering Research*, https://doi.org/10.1007/978-3-319-62217-0_32

functions to components of our systems. Finally, we integrate these components into a true system, testing along the way, before releasing the system for operation.

This says what we need to do, but not how to do it. A formal foundation would supplement this framework with concrete tools and formal methods for accomplishing each step. Our goal in this paper is to propose a candidate approach for such a foundation, based on a branch of mathematics called category theory (CT).

We should mention some prior work associating CT and systems engineering. For example, CT is listed as a foundational approach in the Systems Engineering Body of Knowledge (SEBOK, [1]), although there is little detail associated with the entry. More substantively, Arbib & Manes [2] studied applications of CT in systems control in the 1970s. This work was largely stymied by the unfamiliarity of categorical ideas and the lack of good tools for implementing them (on which we will have more to say in the conclusion).

CT is the mathematical theory of abstract processes, and as such it encompasses both physics and computation. This alone makes it a good candidate for foundational work on modern systems. As we proceed, we will also argue for other virtues including expressivity, precision, universality, and modularity among others.

To make our argument, we will trace through the classic V-model of systems engineering, demonstrating along the way how CT might apply at each step in the process. We have chosen the V-model not for validity (it oversimplifies) but merely for familiarity.

In tracing the V, we hope to accomplish two things. First, we aim to demonstrate the range of categorical methods in order to demonstrate that CT might provide a holistic foundation for systems engineering. Second, and more important, we hope to introduce systems engineers to the language and methods of CT, and pique the interest of the systems engineering community to investigate further. Our hope is that 1 day soon, this paper might serve as the preface to a much deeper study that systems engineers and category theorists might write together.

32.2 Conceptualization

The first role for CT in systems engineering is as a precise technical language in which to express and analyze models of systems information, ranging from theoretical predictions to raw data. The key feature of CT in this respect is its abstraction. We can form categorical models from graphs, from logical ontologies, from dynamical systems and more, and we can use categorical language to analyze the relationships and interactions between these. To get a sense of what this looks like, we will model some simple system architectures and the relationships between them.

The categorical model for an abstract network is remarkably simple:

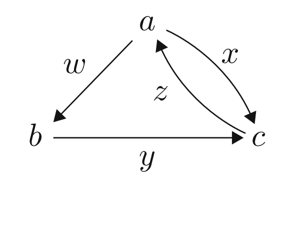
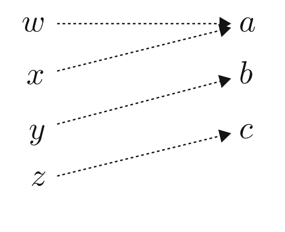
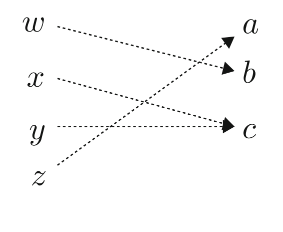
		
A network instance	Source function	Target function

Fig. 32.1 Network as an $\mathcal{N}$-instance

$$\mathcal{N} = \left\{ \texttt{Channel} \overset{\texttt{source}}{\underset{\texttt{target}}{\rightrightarrows}} \texttt{Node} \right\} \tag{32.1}$$

The first thing to observe is that a category contains two types of entities, called *objects* and *arrows*. Intuitively, we think of these assets and functions, though they are abstract in the model itself. An *instance* of the model replaces abstract objects and arrows with concrete sets and functions. It is not hard to see that any network can be encoded as an instance of $\mathcal{N}$, as in Fig. 32.1.

The key differences between categories and directed graphs are the construction principles, which allow us to combine the elements of our models. Foremost among these construction principles is arrow *composition*; whenever we are given sequential arrows $A \xrightarrow{f} B \xrightarrow{g} C$, we can build a new arrow $f\,.\,g : A \to C$. Another way to think of this is, when we draw categories as directed graphs, the arrows include *paths* of edges as well as individual arcs. We also allow paths of length 0, called *identities*.

To see why this is useful, consider the following simple model for a hierarchy of depth $\leq n$:

$$\mathcal{H} = \left\{ 1 \xrightarrow{\texttt{root}} \texttt{Node} \circlearrowleft \texttt{parent} \;\middle|\; \texttt{parent}^n = \texttt{const.root} \right\} \tag{32.2}$$

Here, the primary structure is the self-arrow `parent:Node→Node`, which sends each node to the level above it in the hierarchy. By composing `parent` with itself, we can trace our way up the hierarchy from any node.

By itself, this is too flexible. There is nothing to ensure that all nodes are part of the *same* hierarchy and, even worse, our "hierarchy" might contain loops! We can eliminate these worries by demanding that the `parent` map is "eventually constant": after n repetitions, every node ends up at the same place. This involves two ingredients: a *construction* and a *path equation*.

Categorical constructions generalize most set theoretic operations such as unions, intersections, and Cartesian products. The terminal object 1 stands in for a singleton set, and allows us to express the notion of a constant value

$$\begin{tikzcd}
\text{Channel} \arrow[r, "\text{channel}"] \arrow[d, shift right, "\text{source}"'] \arrow[d, shift left, "\text{target}"] & \Gamma+\Delta \arrow[d, shift right, "\text{id/id}"'] \arrow[d, shift left, "\text{id/parent}"] \\
\text{Node} \arrow[r, "\text{layer}"'] & \text{Layer} \arrow[loop right, "\text{parent}"]
\end{tikzcd}$$

$\Gamma = \Delta =$ Layer

source.layer = channel.id/id

target.layer = channel.id/parent

Fig. 32.2 Categorical model for layered architectures

`root∈Node`. The path equation $parent^n = const.root$ forces the *n*th parent of any node to equal `root`, ensuring a single hierarchy with no loops.

A more interesting example is the layered architecture Λ (Fig. 32.2), in which channels must conform to a hierarchy of layers. Here, the path equations constrain where channels may occur, while the + and / constructions express the fact that channels may form either between layers (Γ) or within a layer (Δ).

All of these models are fairly trivial. The main point is that the sorts of class modeling which systems engineers already do is not too far away from a precise formal language. By carefully modeling our concepts at the early stages of systems engineering, we can express requirements more precisely, identify misconceptions and inconsistencies, and establish concrete domain-specific languages. Best of all, we get both intuitive graphical presentations like those found in UML/SysML class diagrams without sacrificing the semantic precision associated with OWL and other formal approaches to ontology.

CT also goes beyond these existing languages. A *functor* is a mapping between categories; it sends object to objects and arrows to (paths of) arrows, without changing the effects of composition. These maps, along with other constructions like colimits and natural transformations, allow us to explicitly identify and represent the relationships between individual categorical models, thereby linking them into larger networks. This allows semantic ontologies to emerge organically from the bottom-up, grounded in practice, in contrast to "upper ontology" approach (e.g., the Basic Formal Ontology [3]), which tries to impose semantic structure from the top down.

A simple example is the idea that a hierarchy is a special type of network. This fact can be formalized as a functor $H : \mathcal{N} \to \mathcal{H}$. To define H we ask, for each component of $\mathcal{N}$, what plays an analogous role in $\mathcal{H}$? The translation for `Node` is clear. In the hierarchy, we have one channel for each node, so `Channel` also maps to the same object `Node`. Since each channel maps from a node to its parent, `target` corresponds with `parent` and `source` with the identity (zero-length path). Putting it all together, we have the functor depicted in Fig. 32.3a. Similarly, we can identify one hierarchy (of layers Λ) and two networks (of channels $\mathcal{C}$ and layers Λ') in the layer architecture, corresponding to the four functors in Fig. 32.3b. We even have a path equation—$H.L = L'$—which acknowledges that the network of layers in Λ is just the same as the network in $\mathcal{H}$ which is constructed from the hierarchy in Λ.

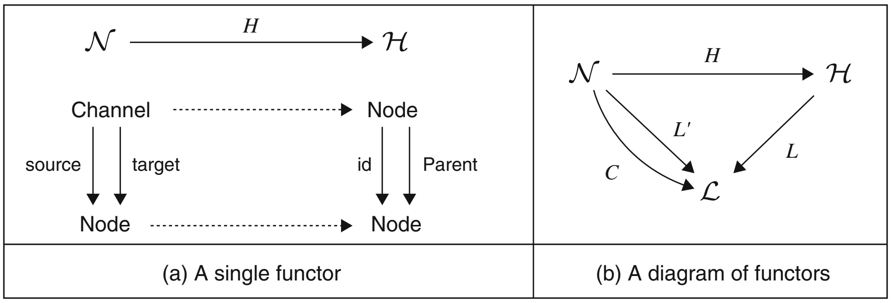


Fig. 32.3 Functors translate between categorical models (**a**) A single functor (**b**) A diagram of functors

The stylized models and relationships presented here are fairly trivial, but the general method of categorical modeling is quite powerful. By varying the constructions, we allow ourselves to use CT modeling range in expressiveness from simple equations to full higher-order logic [12]. For more thorough introductions to categorical modeling, see ref. [23] or [10]. The main thing to remember is that categorical methods provide tools for expressing *and relating* our formal models.

32.3 Decomposition

In the last section, we met all the essential elements of category theory—objects and arrows, composition, identities—except one: the associativity axiom. Given a sequence of three composable arrows $A \xrightarrow{f} B \xrightarrow{g} C \xrightarrow{h} D$, we could first compose at B and then at C, or vice versa. Both should yield the same result: $(f\,.\,g)\,.\,h = f\,.\,(g\,.\,h)$. When applied to processes, this axiom is so obvious, it is difficult to express in English:

Doing *f* and then *g*, and then doing *h*
is the same as
doing *f*, and then doing *g* and then *h*.

Because of this, there is no need to keep track of parentheses when we compose arrows.

This allows us to describe complex processes based on only two pieces of information: (i) the descriptions of simpler subprocesses and (ii) the way they were chained together. Of course, systems engineers know that complex emergent phenomena may arise from simple subprocesses. This does *not* mean that compositional, categorical mathematics does not apply. Instead, it means that the compositional representations of such systems may require greater complexity than the naïve models we might produce from scratch. By demanding compositionality from the outset, we are forced to build interaction into our models from the ground up!

One important step in this direction is to generalize the sorts of composition that we allow. In fact, there are *many* different flavors of category theory, each of which supports a different notion of composition. The plain categories that we met in the last section allow only unary (single-input) processes and serial composition. Some varieties like groups, which formalize the mathematics of symmetry, restrict ordinary categories to obtain simpler structures. Others like process categories and operads add in additional construction principles like parallel composition and multiple input/output. Through these constructions, categories axiomatize the most fundamental concepts in systems engineering: resources and processes [7].

All of these share a common theme of composition and associativity. For groups, this allows us to describe the way that arbitrary rigid motions can be decomposed into translations and rotations. More generally, this allows us to express complicated structures in terms of smaller and simpler pieces. It can also help to show when a chain of complicated operations has a simple and predictable outcome.

Process categories, which are embody the mathematical structure of multi-resource functional decomposition [4, 7]. In the mathematical literature, these are often referred to as "traced symmetric monoidal categories," but we feel that this nomenclature is too imposing given their simplicity and importance. One particularly nice feature of these structures is that process categories support a graphical syntax called *string diagrams* like the one in Fig. 32.4. Completely formal and technically precise, these diagrams are nevertheless as intuitive and easy-to-read as flow charts.

Where string diagrams represent process flows, another class of structures called *operads* formalizes the notion of a parts decomposition [21]. In an operad, the objects are interfaces and the arrows are "wiring diagrams" which connect a set of small interfaces into one larger component. Here, associativity says that there is only one meaning for the phrase "a system of systems of systems."

These representations make it easier to talk about relationships across scale. Some or all of the subprocesses in the Fig. 32.4 will have their own process decompositions. The only substantive constraint on these decompositions is that they have the appropriate input and output strings. This leaves us with one high-level categorical model Π for the entire process and several low-level models Θ_i for the individual subprocesses.

To express the relationship between these, we first combine the low-level pieces into a single aggregate model $\Theta = \bigoplus_i \Theta_i$. This involves an operation called a *colimit* which generalizes set-theoretic unions; building them requires explicitly representing the overlap between different models. Once we build the aggregate model, we can then define a functor $\Pi \to \Theta$ which essentially pastes copies of the smaller diagrams Θ_i into the appropriate bubbles from Π. This identifies an explicit model for the total high-level process Π inside the aggregate low-level model Θ. Furthermore, we can also allow multiple decompositions for a given subprocess, providing a framework for modularity and versioning.

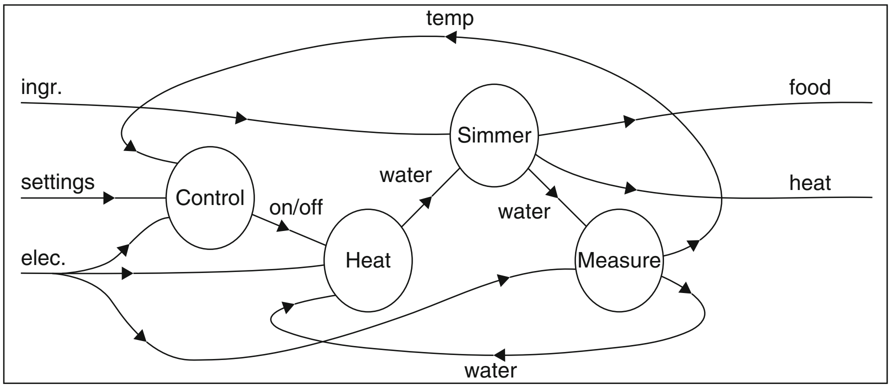


Fig. 32.4 Process decomposition as a string diagram

32.4 Realization

During realization, we turn our abstract models into concrete realizations. In spirit, the relationship between these two is analogous to the that between the logician's notions of syntax and semantics. Roughly speaking, syntax is what we say and semantics is what we mean, or what we are talking about. Models are like syntax: they describe how a product or system is supposed to work in terms of both structure (decomposition and component interaction) and behavior (requirement and verification specifications). Attaching semantics to these models means assigning each syntactic component to some sort of concrete entity, in a way that mirrors the structure and behavior of the model.

Ultimately, these concrete entities will be physical components and functioning source code, but before we reach that point, we must pass through many other, more abstract semantics. These might range from the formal verification of a critical algorithm to a stochastic model of user behavior, but most have some flavor of simulation. The motivating example to keep in mind is the simulation of a system in terms of (discrete, continuous, or hybrid) dynamical systems [15].

The key feature of the logician's semantics is compositionality: if we want to determine the truth of a complex logical formula, it is enough to look at the truth values of its subformulas. This might seem to fail for a given dynamical system: just because each component of my system is safe in isolation hardly guarantees safety of the composite system. Doesn't the existence of emergent phenomena mean that the behavior of a complex system is *not* determined by the behavior of its components? This misunderstanding rests on a conflation of two distinct notions of "behavior."

We can think of system behavior as a path through some high-dimensional state space; component behavior is the projection of this path onto the subspace of component parameters. The problem is that component dynamics in isolation trace out different paths than the projected system dynamics would. This is why component safety in isolation does not entail system safety, even for the same component metrics. This also means that there is no hope of composing individual component behaviors to derive system behavior.

However, dynamical models, the differential equations which generate these paths, *are* composable: we can derive the dynamical equations of a system from the dynamics of its components [24]. The formula for this derivation will, of course, depend on how the components are connected to one another. Each diagram like the one in Fig. 32.4 generates its own formula. CT structures this relationship, making the requirements of compositionality explicit through the language of categories and functors.

Logical semantics involves three main elements: (i) a syntactic model to be interpreted, (ii) an assignment of syntactic elements to semantic objects, and (iii) a satisfaction relation which determines whether this assignment meets the requirements of the model. However, traditional logic operates in a fixed context of sets and functions (deterministic semantics), while CT broadens this to allow stochastic semantics, dynamical semantics, and more. Thus, categorical semantics adds one further element, (iv) a universe of semantic entities.

This approach relies on an important though informal distinction in CT between smaller, "syntactic" categories and larger, "semantic" categories. Syntactic categories are like the architectural models described from Sect. 1, built directly from graphs (generators), path equations (relations), and categorical structure (constructions).

Semantic categories instead use some other formalism, like set theory or matrix algebra, to define the objects and arrows of a category directly. The prototypical example is the category of sets and functions, denoted **Sets**, where composition (and hence path equations) is computed explicitly in terms of the rule $f\,.\,g(x) = g(f(x))$. Many other semantic categories like **Graph** (graphs and homomorphisms) and **Vect** (vector spaces and linear maps) can be constructed from set theoretic entities.

Once we adopt this viewpoint, the relationship between syntax and semantics can be represented as a functor from one type of category to the other. We have already seen one example of this approach, in Fig. 32.1, where we described a network instance in terms of a pair of functions. This is exactly the same as a functor $I : \mathcal{N} \to$ **Sets**: we map objects of $\mathcal{N}$ to objects of **Sets** and arrows of $\mathcal{N}$ to arrows of **Sets** (i.e., to sets and functions).

The satisfaction relation for the semantic interpretation is determined by the preservation of categorical structure. A good example is the coproduct "+," used in our model for the layered architecture Λ (Fig. 32.3). Not all functors $\Lambda \to$ **Sets** are semantically valid, only those which map the abstract coproduct $\Gamma + \Delta \in \Lambda$ to a concrete coproduct (disjoint union) in **Sets**. We say that a model of Λ should preserve coproducts. Implicit in any categorical model is a minimal set of construction principles required to preserve full semantics.

Once we recognize that the traditional (logical) interpretations for a model $\mathcal{M}$ are the structure-preserving functors $\mathcal{M} \to$ **Sets**, we are in an easy position to generalize to a much wider array of semantics. We have explicitly identified the necessary structural context (e.g., coproducts) $\mathcal{M}$, so we can replace **Sets** by any other category which has these same features. We can use a category **Dyn** whose objects are dynamical systems; a functor $\mathcal{M} \rightarrow$ **Dyn** provides dynamical semantics. There is a category **Prob** whose arrows are probabilistic mappings; a functor $\mathcal{M} \rightarrow$ **Prob** describes stochastic semantics for $\mathcal{M}$. There is a computational category **Type** where arrows are algorithms; functors $\mathcal{M} \rightarrow$ **Type** provide computational interpretations for $\mathcal{M}$. We can often compose these, for example, mapping a model to a dynamical system, and then mapping this to a computational simulation. Sometimes we can even mix semantics together, so that in Fig. 32.4, we could give dynamical models for `Heat` and `Simmer`, a computational model of `Control` and a stochastic `Measure`, and compose these to give a hybrid dynamical model for the whole system.

32.5 Integration

The main role of our models in system integration is to collect and manage the tremendous amount of structured data collected and analyzed during the integration process. This data is necessarily heterogeneous, multiscale, and dispersed across many models and experts. Categorical models have several nice features which can support the federation of this data.

First of all, we can regard a finite syntactic category $\mathcal{M}$ (like one of the architectural models in Sect. 1) as a database schema [14, 19, 20]. Roughly speaking, the objects are tables and the arrows are foreign keys. This means that we can use the models already produced during conceptualization and decomposition to store the data generated during integration. Formally, this depends on the functorial semantics discussed in the previous section; we can think of an instance of the database as a functor $I : \mathcal{M} \rightarrow$ **Sets** mapping each table to a set of rows. Notice that this approach automatically ties the data that we produce to our semantic models.

A more significant challenge is the dispersion of data across many engineers using many different models. In order to build a holistic picture of our system, we need some way of putting models together and aggregating the data they contain. The CT approach involves a categorical construction called a *colimit*, together with an additional twist.

A colimit is a categorical construction that generalizes unions, allowing us to build new objects by gluing together old ones. For example, any graph can be constructed using colimits by gluing edges together at nodes. To integrate two objects using a colimit, we first explicitly identify their overlap as a third object, along with two maps embedding the overlap into each component. Given this data, the colimit construction then produces a fourth object together with two maps which embed the original components into the new object. See Fig. 32.5a.

The twist is that, instead of looking at categorical constructions inside our models, now we are interested in performing colimits *with* our models. This approach depends on the fact that CT is self-referential: the methods of CT can

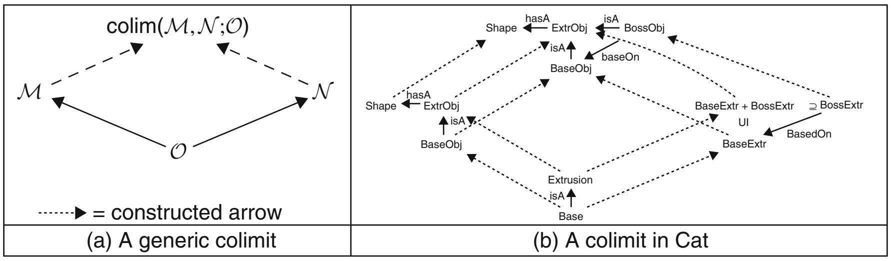


Fig. 32.5 The colimit construction (**a**) A generic colimit (**b**) A colimit in Cat

be applied to study categories themselves. In particular, there is a semantic category **Cat** whose objects are categories and whose arrows are functors. Colimits in this and related semantic contexts can be used to define model integration. A very simple example is given in Fig. 32.5b.

In fact, we can form colimits from any number of components, so long as we accurately represent their overlaps (and overlaps of overlaps, etc.), providing a scheme for wider integrations. However, representing all those overlaps may be inefficient. Another alternative is to integrate serially, adding in one new model at a time. CT provides us with a language to state and prove that either approach is valid, and that the two options will yield equivalent results [25].

As for heterogeneity, CT constructions called *sheaves* have recently been proposed as "the canonical datastructure for sensor integration" [18]. The main idea is that when different sensors capture overlapping information, it must be restricted or transformed before it can be compared. In the simplest example, to identify overlapping images, we must first crop to their common ground (restriction) before comparing the results. A simplistic algorithm would ask for perfect agreement on the restriction, but a more sophisticated integration might allow small differences in shading or perspective (transformation). We can also compare different types of information, so long as we can project them to a common context; we might match up audio and video by translating both to time series and looking for common patterns. CT provides the language and spells out the requirements for translating between contexts in this way.

Finally, by mixing colimits with functors, we can connect our models across layers of abstraction [6]. Suppose that $\mathcal{H}$ is a model one level of abstraction above that of $\mathcal{M}$ and $\mathcal{N}$ in Fig. 32.5. Both $\mathcal{M}$ and $\mathcal{N}$ are more detailed than $\mathcal{H}$, but each only covers half the range. When we put them together, though, they *do* cover the same range: every entity of $\mathcal{H}$ can be defined by mixing structures from $\mathcal{M}$ and from $\mathcal{N}$. Formally, this means that we can construct a refinement functor $\mathcal{H} \to \mathrm{colim}(\mathcal{M}, \mathcal{N}; \mathcal{O})$ which tells us how to compute high-level characteristics in terms of low-level ones, helping to trace high-level requirements to low-level performance.

32.6 Operation

In operation, systems are never static. Components fail and need to be replaced. New models and versions require tweaks to existing production and control system. New technology or regulation changes the environment in which our systems operate. Because of this, it is critical that our models should be relatively easy to maintain and update. Here again, categorical methods have some nice features which recommend them.

One significant challenge in updating a model is that we must take existing data attached to the original model and shift it over to the new one. Thinking of our models as domain-specific languages, we must translate our data from one language to another. These processes are often messy and ad hoc, but categorical constructions can help to structure them.

As we mentioned in the last section, a class-type categorical model $\mathcal{N}$ like those discussed in Sect. 1 can be translated more-or-less directly into database schemas [14, 19, 20] where objects are tables and arrows are foreign keys. An instance of the database is a functor $\mathcal{N} \to \mathbf{Sets}$ which sends each abstract table to a concrete set of rows. By generating our data stores directly from models, our data are automatically tied to its semantics.

We can then use functors to formalize the relationship between old and new models. This will provide a dictionary to guide our translation. Moreover, expressing the transformations in these terms can help to organize and explain certain inevitable features of this process.

A good example is the phenomenon of duality between models and data. A meticulous reader will have noted that, in the discussion of architectural models, we said that "every hierarchy is a special kind of network," but then proceeded to define a functor $\mathcal{N} \to \mathcal{H}$. The direction has reversed!

The categorical formulation explains this fact: given a functor $\mathcal{N} \to \mathcal{H}$ and an instance $\mathcal{H} \to \mathbf{Sets}$, we can compose these at $\mathcal{H}$ to obtain an instance $\mathcal{N} \to \mathbf{Sets}$. So every functor between syntactic models defines a mapping of instances *in the opposite direction*. We might call this operation model restriction or projection, and categorically speaking it is simply composition.

While composition allows us to restrict data backward along a functor, subtler and more significant constructions called *Kan extensions* allow us to push data in the same direction as a functor [20]. In many cases, data demanded by the new model will be unavailable in the old; in others, we may split one concept into two, or vice versa. In all of these cases, Kan extensions provide explicit instructions for building a "best approximation" to the old data, subordinate to the new schema.

Remarkably, the same operation of Kan extension can also be used to encode quantification in formal logic [17] and periodic states in dynamical systems [15]. This points to a critically important aspect of categorical methods: uniformity. The abstraction of CT allows us to apply the same set of tools to a remarkably diverse set of problems and circumstances.

This can be problematic for beginners: even simple applications of CT may require learning several abstract constructions. Why bother, when there are easier solutions to this problem or that? The value of the CT approach only becomes apparent for more substantive problems, where the same familiar tools can still be applied.

Another nice property of categorical models is modularity, which is supported by the fact that the colimit construction is a functor. Suppose, for example, that we extend one of the models in Fig. 32.5a via a functor $\mathcal{N} \to \mathcal{N}'$. A categorical construction principle for the colimit then guarantees that we can build a new map $\mathrm{colim}(\mathcal{M}, \mathcal{N}; \mathcal{O}) \to \mathrm{colim}\left(\mathcal{M}, \mathcal{N}'; \mathcal{O}\right)$. This allows us to update domain-specific models locally and then lift these changes to a global context.

More generally, the category theoretic property of naturality (over the diagram of the colimit) encodes the restrictions which must be satisfied if updates to multiple components are to be consistent with one another. Other categorical constructions called *fibrations* have been useful in formalizing more general bidirectional transformations, where updates may not be consistent with one another [9, 13]. In fact, the elucidation of this concept of naturality was the motivating goal in the original development of CT; categories and functors were merely the supporting concepts which underpin "natural transformations" [11].

Our discussion here has tried to indicate the potential breadth of categorical analysis. In so doing, we have sacrificed depth in return. There is much more to be said.

32.7 Conclusion

One by one, the elements of category theory may not seem so impressive. We already have OWL for representing semantic information, and good tools for interacting with databases. The UML/SysML language family allows us to build graphical models and translate them into code stubs for programming. Modelica and other modeling languages allow us to describe component-based decompositions and link these to dynamical simulations. R and other software provide tools for statistical modeling.

The real value of CT is that it provides a context in which all of these can interact, and a rigorous language for defining and analyzing those interactions. Now, we have a chance to formalize entire toolchains and workflows: we can agree on a graphical model, produce from it a semantic (logical) model, and populate it with data from an existing schema. We can use that data to derive a dynamical model, and transform this into a computational simulation before piping the results to statistical software for analysis. This entire process can be structured by categorical models.

This indicates why systems engineering offers an ideal test bed for the emerging discipline of applied category theory. First, there is no avoiding the need to employ formal methods from multiple disciplines. The details of our system exist at different scales and layers of abstraction. The need to interface between many groups and researchers generates many demands: precise language to prevent misunderstanding, intuitive (e.g., graphical) representations for easy communication, and structural modularity for putting these pieces together.

Today, CT can supply plausible suggestions for meeting all of these requirements and more. However, much work is required to turn this promise into practice. We can identify at least two important obstacles which have stymied the growth of applied category theory.

First of these is CT's learning curve, which is undeniably steep, but has become more gentle in recent years. New textbooks [16, 22] targeted at scientists and undergraduates have made the mathematical ideas more accessible. New applications in areas like chemistry [7], electrical engineering [5], and machine learning [8] have broadened the base of examples to more concrete, real-world problems.

A more substantial obstacle is tool support. Today, CT can solve many problems at the conceptual level, but there are few good tools for implementing those solutions. Outside of functional programming (one of the major successes of CT) most software is academic, and it is neither simple enough nor powerful enough to address system-scale demands. Addressing this deficiency will require substantial funding and a concerted effort to bring together mathematicians with domain experts to attack complex, real-world problems.

Fortunately, this requirement is less daunting than it seems. Because CT generalizes many other formalisms, we should be able to use existing tools to solve categorically formulated problems. By turning a category into a logical theory, we can use an OWL theorem prover for validation. To analyze the behavior of a functional model, we can derive a Petri net for simulation. By projecting our categorical models back into existing formalisms, we can piggyback on existing tools and methods. The results of these analyses can then be lifted back to the categorical level for a holistic appraisal.

We envision an open, CT-based platform for information modeling and analysis. The platform should support modules for the various CT constructions (e.g., functors and colimits) and translations (OWL, SQL, and petri nets), which could then be assembled on a case-by-case basis to address specific problems. In the long run, such a platform would be applicable across many domains, but to get there, we first need to drill down and provide a proof of concept. Systems engineering is the perfect candidate.

Disclaimer Any mention of commercial products within NIST web pages is for information only; it does not imply recommendation or endorsement by NIST.

References

1. Abran A, Moore JW, Bourque P, Dupuise R, Tripp LL (2004) Software engineering body of knowledge. IEEE Computer Society, New York
2. Arbib M, Manes G (1974) Foundations of system theory: decomposable systems. Automatica 10(3):285–302
3. Arp R, Smith B, Spear AD (2015) Building ontologies with basic formal ontology. MIT Press, Cambridge
4. Baez J, Stay M (2011) Physics, topology, logic and computation: a Rosetta stone. In: Coecke B (ed) New structures for physics. Springer, Heidelberg, pp 95–168
5. Baez J, Fong B (2015) A compositional framework for passive linear networks. *arXiv preprint* 2015:1504.05625
6. Breiner S, Subrahmanian E, Jones A (2016) Categorical models for process planning. Under review: *Computers and Industry*
7. Coecke B, Fritz T, Spekkens RW (2014) A mathematical theory of resources. Inf Comput 250:59–86
8. Culbertson J, Sturtz K (2013) Bayesian machine learning via category theory. *arXiv preprint* 2013:1312.1445
9. Diskin Z (2008) Algebraic models for bidirectional model synchronization. In: Czarnecki K et al (eds) International conference on model driven engineering languages and systems. Springer, Berlin, pp 21–36
10. Diskin Z, Maibaum T (2014) Category theory and model-driven engineering: from formal semantics to design patterns and beyond. In: Cretu LG, Dumitriu F (eds) Model-driven engineering of information systems: principles, techniques, and practice. Apple, Toronto, pp 173–206
11. Eilenberg S, Mac LS (1945) General theory of natural equivalences. Trans Am Math Soc 58 (2):231–294
12. Jacobs B (1999) Categorical logic and type theory. Elsevier, New York
13. Johnson M, Rosebrugh R, Wood RJ (2012) Lenses, brations and universal translations. Math Struct Comput Sci 22(01):25–42
14. Johnson M, Rosebrugh R, Wood RJ (2002) Entity-relationship-attribute designs and sketches. Theory Appl Categ 10(3):94–112
15. Lawvere FW (1986) Taking categories seriously. Revista Colombiana de Matematicas XX:147–178
16. Lawvere FW, Schanuel SH (2009) Conceptual mathematics: a first introduction to categories. Cambridge University Press, Cambridge
17. MacLane S, Moerdijk I (2012) Sheaves in geometry and logic: a first introduction to topos theory. Springer Science & Business Media, New York
18. Robinson M (2016). Sheaves are the canonical data structure for sensor integration. *arXiv preprint* 2016:1603.01446
19. Rosebrugh R, Wood RJ (1992) Relational databases and indexed categories. In: Seely RAG (ed) *Proceedings of the International Category Theory Meeting 1991*, vol 13. Canadian Mathematical Society, Providence, pp 391–407
20. Spivak DI (2012) Functorial data migration. Inf Comput 217:31–51
21. Spivak DI (2013) The operad of wiring diagrams: formalizing a graphical language for databases, recursion, and plug-and-play circuits. *arXiv preprint* 2013:1305.0297
22. Spivak DI (2014) Category theory for the sciences. MIT Press, Cambridge

23. Spivak DI, Kent RE (2012) Ologs: a categorical framework for knowledge representation. PLoS One 7(1):e24274
24. Spivak DI, Vasilakopoulou C, Schultz P (2016). Dynamical systems and sheaves. *arXiv preprint* 2016:1609.08086
25. Wisnesky R, Breiner S, Jones A, Spivak DI, Subrahmanian E (In press) Using category theory to facilitate multiple manufacturing service database integration. J Comput Inf Sci Eng

Part V
System Architecture and Complexity

Chapter 33
A Facilitated Expert-Based Approach to Architecting “Prizeable” Complex Systems

Zoe Szajnfarber and Ademir Vrolijk

Abstract This study builds on the premise that open innovation methods can be effectively applied to more complex engineered systems through a particular kind of decomposition, one that decouples parts that are suitable for (1) distant expert search, (2) sampling from the right tail, and (3) force multiplying. This paper develops and demonstrates a method that leverages a facilitated expert workshop to elicit those kinds of “prizeable” problems. Our research context – the NASA Asteroid Grand Challenge – had previously suffered from the perception that there was no meaningful role for open innovation methods to play since the physics was such that observation is dominated by contributions from institutional players. However, through our workshop, we both demonstrated that prizeable subproblems exist – even in this highly complex system – and that the proposed approach is capable of eliciting them. The paper concludes by reflecting on the implications of this exercise for open innovation in general.

Keywords Open innovation • System architecting • Decomposition • Decontextualization • Recontextualization • Complex systems • Planetary defense

33.1 Introduction

Prize competitions are increasingly being touted as the solution to the stagnation of our national innovation economy. For example, the America COMPETES Reauthorization Act of 2010 encourages the use of open innovation (OI) methods to “spur innovation, solve tough problems, and advance their core missions” [1], quoting America COMPETES, [2]. These methods, which include citizen science, prizes and challenges, and crowdsourcing activities, are part of a broader trend to use OI as part of an organization’s innovation toolkit [1, 3, 4].

While the approach has been demonstrated to be highly effective on certain kinds of problems (e.g., XPRIZE and NASA Tournament Lab), in their current form, OI methods are not equally effective for all types of systems and problems

Z. Szajnfarber (✉) • A. Vrolijk
The George Washington University, Washington, DC, USA
e-mail: zszajnfa@gwu.edu; avrolijk@gwu.edu

A.M. Madni et al. (eds.), *Disciplinary Convergence in Systems Engineering Research*, https://doi.org/10.1007/978-3-319-62217-0_33

[5–7]. Should the agencies apply OI methods too broadly in advance of guiding theory, they could run the risk of dismissing OI methods before their value and potential can be captured. Currently, there is a strong skepticism among engineering-oriented agencies as to whether OI methods can be applied to their physically integral system design problems. This skepticism also pervaded the NASA's Asteroid Grand Challenge (AGC), the context for this work.

In previous work, we have argued that a productive path to extending the applicability of OI methods involves decomposing the full problem to isolate subproblems that are more "prizeable" [8]. In other words, we advocate a strategy of architecting prizeable systems that keep some parts inside as the domain of disciplinary experts while "opening" particular pieces that are most amenable to input from the "crowd." To extend that line of reasoning, this paper does three things. First, we present a method for facilitating expert problem formulation to enable more effective use of OI for complex systems. Second, we describe how this method was tested through implementation in the context of asteroid detection (a part of NASA's AGC). We will show that the method can serve to both mitigate stakeholder resistance to using OI tools and also identify new and productive prizeable parts. Finally, we will discuss lessons learned from this implementation experiment and avenues for future research.

33.2 Theoretical Basis for the Approach

33.2.1 Why Sampling "Outside" the Usual Suspects Can Yield Better Results

There are two basic arguments for why outsiders (novices) can do better than internal experts: right-tail sampling and distant experts.

The first is a basic probability argument. Recognizing that while experts are individually, and on average, better than novices, novice (externally derived) solutions can still improve solution quality. This is enabled by ex post selection in innovation tournaments, allowing the seeking organization to pick from the best of the novice solutions – the "right tail" in a distribution of quality. As long as this right tail extends beyond the mean of the expert distribution, the best response from a tournament of novices will often do better than what is effectively a single "draw" by an internal expert.

The second focuses on how problem solving is done. Internal experts tend to be cognitively entrenched according to their expertise [9]. This means that they are likely to search locally [10], coming to solutions similar to the incumbent product that will work well in their context. Though local searches have the advantage of proceeding quickly, when a radically different approach is needed, internal experts are less likely to find it. Outsiders, on the other hand, have the advantage of being unencumbered by the "usual way of doing things." They may see the problem differently [11], and this may lead to new solution paths [9, 12, 13].

A specific class of outsiders, named distant experts – who possess distant knowledge or are technically marginal [13–16] – can be particularly valuable in this respect. Though one might not expect their expertise to be relevant to the focal domain, they do have expertise relevant to the fundamental problem. Viewing the problem from their (different) disciplinary perspective, their approach can lead to frame-breaking solutions. A classic example is the Victoria Secret mechanical wing designer, who brought a novel solution to the design of NASA's astronaut gloves [17].

33.2.2 Why OI Struggles When Applied to the Whole Complex System

The above arguments explain why OI methods have been successful in solving certain kinds of problems. Unfortunately, both mechanisms break down when applied to increasingly complex systems. First, complex systems tend to draw on multiple "naturally" uncorrelated skills. For example, designing an aircraft requires expertise in controls, structures, and aerodynamics. While many domains build expertise in each of those disciplines individually (e.g., new ideas for the aircraft body may come from automakers, ship builders, or even building design), it is unlikely to find a single solver with cross-training in all three areas unless they were trained as an aerospace engineer. To generalize this, Fig. 33.1 shows that while you might expect to achieve extreme values on each subsystem individually, the joint distribution over this skill/subsystem mapping tends to separate experts with relevant training from those without. As a result, it is unlikely for even a right-tail novice solution to compete with expert solutions.

In the same way, the extent of required specialization and co-location of skills tends to limit the likelihood of finding a distant expert. Though distant experts may be difficult to identify generally, one does not necessarily need to know where they are to identify them through a tournament. That said, finding distant experts through a tournament is inversely related to the tightly interconnected skills required, because that relationship tends to reduce the pool of solvers. Specifically, in cases where distant experts have been effective, their relevance was only obvious ex post.

33.2.3 How Decomposing the Complex System Can Help

The act of decomposition monotonically reduces the complexity of the resultant subproblem. This is often done to make complex system design tractable by (a) enabling parts of the problem to be worked independently and in parallel [18–

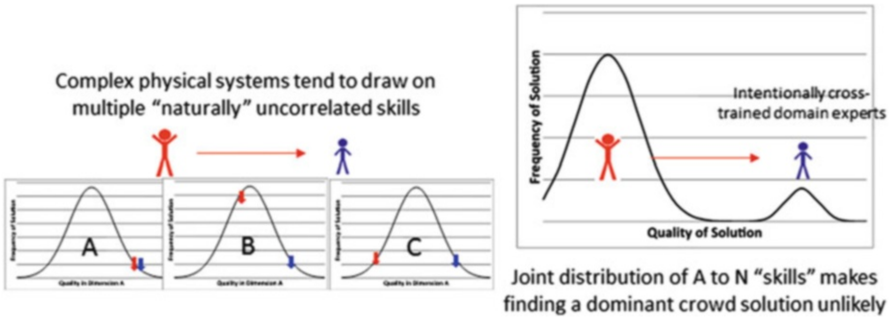

Complex physical systems tend to draw on multiple "naturally" uncorrelated skills
A
B
C
Frequency of Solution
Intentionally cross-trained domain experts
Quality of Solution
Joint distribution of A to N "skills" makes finding a dominant crowd solution unlikely

Fig. 33.1 Challenge of right-tail mechanism applied to whole complex system

20] and (b) reducing the extent of system knowledge required of any single entity [21]. Although generally done within an organization or enterprise, the same logic applies as a strategy for enabling wider participation in system design. Instead of dividing work among departments and subsidiaries, in the context of OI, parts of the work are allocated to the "crowd."

Decomposition works well when there is a good "fit" between the structure of the organization doing the solving and the product being developed [20, 22, 23]. Thus, good decomposition for OI needs to isolate parts of the problem that match the kind of expertise and skills that are nascent in the pool of solvers willing to respond to the prize competition (see mechanisms described in Sect. 33.2.1). It is worth noting that "opening the system" does not mean that all aspects of the system need to be solved externally. A smart decomposition will keep some parts internal while opening up others.

33.2.4 Three Specific "Parts" to Look for

While not necessarily exhaustive, the above discussion leads to at least three types of subproblems that good decomposition should seek to isolate: (1) distant expert search, (2) variability dominates quality, and (3) force multiplier. We define each in terms of the type of problem, the kind of solver being sought, and the process through which OI helps.

Distant Expert Search Defined by a *problem* that is hard on one, or a small number of, dimensions of expertise. The smaller number of dimensions of expertise, the higher the likelihood of finding an external *solver* capable of contributing an extremely good solution. That *solver* will identify with a different domain and will have cultivated their relevant expertise through either a serious hobby interest (in the origin) domain or because the relevant dimension of expertise is also important in their (different) home domain. The *search process* can use either a wide broadcast search of a decontextualized problem or a pre-identification of external populations that may contain the relevant single-dimension expertise.

Variability Dominates Quality Defined by a *problem* where the variance in expert attempts is larger than the difference between the mean of contributions by novices and experts. Here, we expect the minimum required capability threshold to achieve a meaningful contribution to be relatively low, increasing the solver pool. The goal of the *search process* for this kind of problem is to first solicit inputs from as many *solvers* as possible and then to select the best ex post from the contributions.

Force Multiplier Defined by a *problem* with a low skill threshold, an outcome that can be specified in advance and a high workload. In general, these are problems where algorithms are less effective, e.g., image characterization. Similar to the variability reduction category, the *solvers* do not need to be particularly skilled. In this case, the goal of the *search process* is to get lots of good enough inputs and combine all of them as a single solution.

33.2.5 Why Good Results May Be Poorly Received

The previous sections have focused on how to get high-quality results through on OI approach. Equally important (and difficult) is ensuring that good results will be well received and used as appropriate. Barriers to achieving this goal include (a) the cost of infusing the solution, (b) a change in context between the start and end of the challenge, and (c) internal resistance to external solutions. One high-profile example that showcases (a) and (b) is the Netflix challenge. Although the winning algorithm exceeded expectations, it was never fully implemented because the engineering effort that would have been required to integrate the solution exceeded its projected returns [24, 25]. In addition, Netflix noted a dramatic shift in their business model (DVDs to streaming video), which significantly changed the way customers would rate their titles (invalidating the current algorithms). With respect to (c), prior work has explained this phenomenon in terms of internal stakeholders resisting what they see has an attack on their professional identity. From the perspective of the internal expert, management is suggesting that they ask "Joe Average" for help solving a problem they have spent their whole careers working on [26]. To overcome this challenge, it is necessary to garner buy-in early in the process. Not surprisingly, if the internal team is involved in the decision to seek external contributions on particular parts, they will be more likely to use the results.

33.2.6 Facilitated-Expert Workshop

Based on the above, we formulated and facilitated an expert workshop, designed to identify "prizeable" (sub)problems within a particular context. The workshop led participants to decompose the problem for OI activities, thus maximizing the value of the two core OI mechanisms (discussed in Sect. 33.2.1) by identifying one of the three kinds of problems (discussed in Sect. 33.2.4). Section 33.3 describes the development of the method and its application to NASA's Asteroid Grand Challenge in an integrated way.

33.3 Case Study: Opportunities for OI in Asteroid Detection

In July 2015, we had the opportunity to test the utility of these ideas in practice. As part of the AGC initiative, NASA launched a summer fellowship program with a goal of surfacing frame-breaking ideas for improved asteroid detection. The fellowship would bring together graduate students, with nontraditional disciplinary backgrounds, in an intense, mentored environment.

Our charge was to maximize the likelihood of a successful outcome for the program by (a) identifying a suitable (sub)problem around which the summer program would be focused and (b) providing guidance on the range of disciplines to target. A successful outcome was loosely defined as a technical product (could be concept or artifact) that would (1) enable progress toward the stated goal of 90% detection of potentially hazardous objects (PHO) above 140 m in diameter [27] and (2) provide the summer participants with a meaningful research opportunity while supporting the work of the planetary defense community in this focused research activity [28].

With those goals in mind, we designed a 2-h, 15-person workshop that piggybacked a standing meeting of experts in the field of planetary defense. This section begins by providing brief background on the asteroid detection problem, explains how we structured the workshop, and describes the results of the process. Section 33.4 draws out lessons learned and relates them back to our proposed general process for identifying prizeable parts.

33.3.1 Background: Asteroid Detection and the Asteroid Grand Challenge

In 2013, planetary defense from asteroids became the focus of NASA's first Grand Challenge. Its stated goal was "to find all asteroid threats to human populations and know what to do about them." The challenge hoped to shine light on the problem and mobilize citizen scientists and nontraditional contributors to ramp up detection (finding them) while providing potential solutions to deflection (do something about them). In this paper, we focus exclusively on the detection piece.

While NASA intended the Grand Challenge to complement the ongoing Planetary Science work, initial interactions with that community were met with aggressive resistance – mainly due to their skepticism of the solvers and their professional identity destruction (as described in Sect. 33.2.5). However, there is also a physics-basis for this reticence.[1] Since these types of observations need to be made from outside the Earth's atmosphere (from satellites) or using very large ground-based observatories (on the order of several meters), the capital expense is such that no one not already backed by a large institution can reasonably participate.

Thus, the image that the AGC painted of an army of citizen scientists making backyard discoveries with hobby telescopes was, in the mind of the planetary science community, unrealistic. As such, external, nontraditional, contributions will need to be focused on carefully selected subproblems.

33.3.2 Workshop Structure

The workshop was designed in three parts: (a) a functional decomposition exercise, (b) opportunities for OI exercise, and (c) a solver mapping exercise. The sections that follow describe each exercise and the underlying rationale for structuring them in this way. Recognizing that community buy-in was at least as important as a well-designed problem, we kept the need for buy-in in mind throughout the workshop.

Step 1: Functional Decomposition As explained in Sect. 33.2, good decomposition (from the perspective of enabling meaningful external contributions) needs to isolate particular kinds of subproblems. For the asteroid detection problem, we focused on two kinds: (1) Problems that lend themselves to distant expert search, wherein a deep single-discipline challenge is isolated and the search process seeks to identify a nontraditional player with previously unidentified but highly relevant expertise. That "distant expert" may provide a frame-breaking solution because they will view the problem from their own disciplinary perspective. (2) Problems where a solution path is known but the sheer volume of work limits progress. In

[1]These observational activities require large, multifaceted infrastructure spread across the globe because of (a) the small size of the objects that could pose a threat (both physical size and apparent size due to reflectivity) [29], (b) the multiple observations required to establish the orbit (when observing in optical spectrum) [27, 29], and (c) the limitations of individual platforms in detecting or characterizing a given PHO [29, 30]. With this infrastructure at their disposal, the vast majority of near-Earth object (NEO) discoveries are made by ground-based optical systems (bolstered by space-based observations) [27, 31] with follow-up characterization observations performed by radar telescopes when required. Since the beginning of their discovery and characterization efforts in 1998, detection capabilities have improved from an estimated 90% of all objects larger than 1 km in diameter to 90% of those greater than 140 m. Experts believe that the current discovery rate of approximately 1000 per year [29] is only limited by the observational infrastructure dedicated to the task.

such cases, a well-structured challenge can leverage latent resources in the crowd and that coordinated effort becomes the solution.

Since the AGC team had previously received so much resistance when they suggested potentially “openable” problems, we thought it important to draw potential problems organically from discussion with the experts. However, we also recognized that experts are notoriously poor at seeing problems differently, that existing architectures tend to only make sense for existing solvers, and that there is a clear disincentive to identifying one’s own area of expertise as the challengeable part. Thus, we needed to be careful in how the exercise was framed.

To overcome these challenges, we structured the exercise to take focus off the future solver and instead emphasize inherent functions. We did this in two parts:

Step 1a: Asked experts to identify the functions required to do “detection.” In advance of the exercise, the facilitators wrote many of the system functions on sticky notes. This created a starting point upon which participating experts could add and modify content.

Step 1b: Asked experts to organize functions in process flow. Participants were next asked to order the functions and draw dependencies among them. They did this by placing sticky notes on a whiteboard and drawing flows with whiteboard marker.

While no new information came out of this step (the functional decomposition was fairly well established), it served to initiate all participants to a common language while also provided us a map of the dependencies in the system. The output is shown in Fig. 33.2.

Step 2: Opportunities for OI Step 2 leverages the functional decomposition from Step 1 as a basis for asking the participants to code the level of expertise and resources needed to accomplish the tasks associated with each of the functional elements. Since it is human nature to overestimate the difficulty and importance of one’s own parts of the problem, arriving at a balanced representation of the system requires a careful structuring of the elicitation process. We structured this exercise as a collaborative sorting activity:

Step 2a: We asked experts to transpose the functional elements/measurement types from Step 1 onto colored sticky notes, where the darker the color (red) indicated a higher expertise threshold. This overt coding enabled discussion among participants about the actual difficulty of tasks. We were pleasantly surprised that there was a high level of agreement among experts on the difficulty of the tasks, lending credence to the coding.

Step 2b: Next, experts discussed placement of the sticky notes on the matrix pictured in Fig. 33.3. The horizontal axis captured the method for acquiring information (e.g., lab experiment vs. need for an in situ sample). The vertical axis categorized the extent to which this task was already being done. Unlike step 2a, 2b created some heated discussion. Participants fragmented into groups based on their particular area of expertise and discussed sticky note placement extensively. By the end of the discussion (~30 min), there seemed to be comfort, if not clear consensus on the placement. While not everyone reviewed each of the stickies, experts were

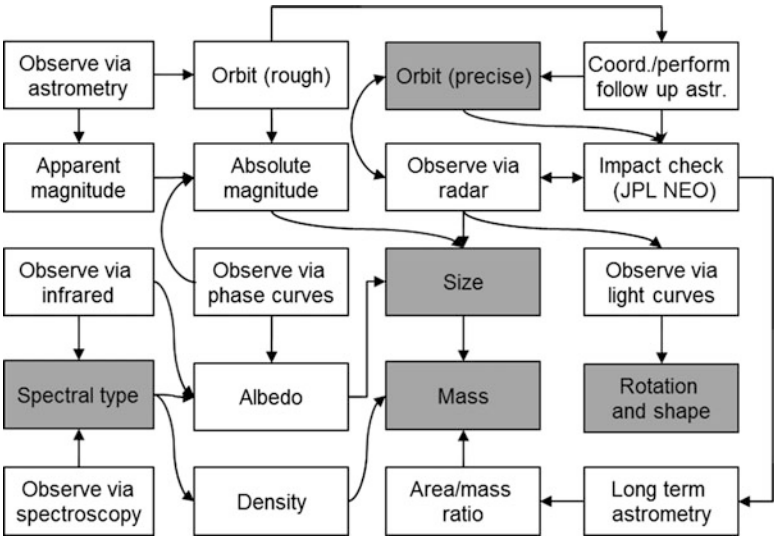


Fig. 33.2 Detection and characterization flowchart (dark denotes end products)

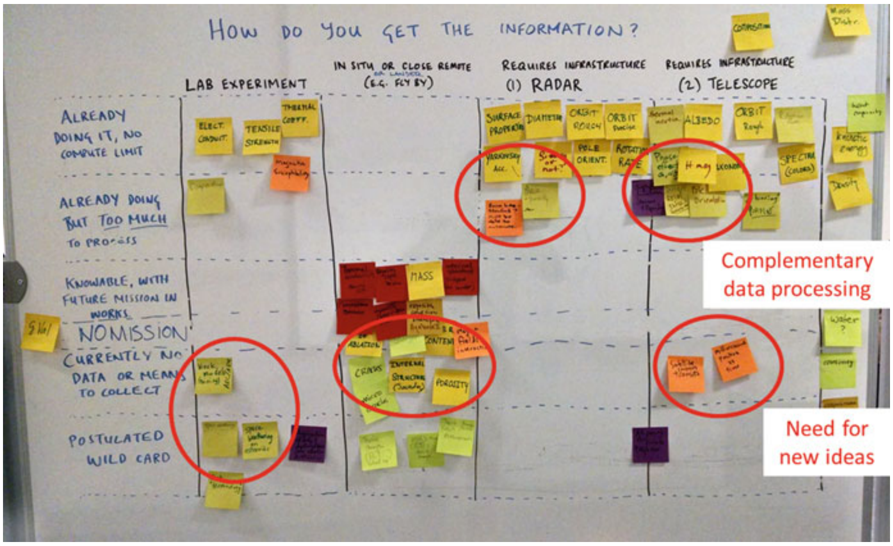

How do you get the information?
Lab experiment
In situ or close remote (e.g. fly by)
Requires infrastructure (1) Radar
Requires infrastructure (2) Telescope
Already doing it, no compute limit
Already doing but too much to process
Knowable, with future mission in works
No mission currently, no data or means to collect
Postulated wild card
Complementary data processing
Need for new ideas

Fig. 33.3 Step 2, map of opportunities for OI

observed to pay close attention to the ones they cared about. By the end of the discussion, there were no objections.

Organizing the discussion map in this way served the dual purpose of (a) focusing the participants on the issues they were interested in (e.g., the kind of new mission that would solve current limitations) and (b) providing the research team with the basis for identifying potential opportunities for external contributions. From our perspective, the x-axis separated out particular barriers to entry, and the y-axis identified gaps. For example, the entire "already doing it, but too much data to realistically process" row identities a potential need for leveraging the efforts of OI solvers. Further, the color coding distinguishes between activities that were coded as red which require a high level of specialized expertise, limiting contributions to internal experts or distant experts, or light yellow which could leverage non-experts at large.

Thus, the large map allowed all participants to contribute the information relevant to them while providing a clear basis for narrowing down the aspects of the problem worthy of future discussion with respect to OI. Specifically, the sticky notes in the "already doing it, but too much to process" row should be examined for potential for complementary processing, and the sticky notes in the "currently no data or means to collect" row identified areas where progress was currently stalled, and there may be more willingness to consider frame-breaking ideas. At minimum, these are areas where external ideas wouldn't be perceived as a direct attack on existing communities.

This process reduced the pool of potential subproblems to about 15, which was still more than could reasonably be discussed in the decontextualization exercise (Step 3). To refine the list further, we incorporated the color code information as well as the process map from Step 1. Systems engineering 101 teaches that clean interfaces facilitate integration. Therefore, we focused on subproblems that had few interdependencies with other subproblems (i.e., loosely coupled). We could have looked for ways to actively decouple subproblems, but at this stage, we wanted to focus on low-hanging fruit.

Finally, since this workshop aimed to support the design of a summer fellowship program (i.e., one particular kind of OI activity), we sorted for medium difficulty tasks that would be appropriate for the kind of participant being recruited. If, on the other hand, we had been interested in running a more traditional challenge, we would likely have focused on the yellow complementary processing tasks and/or the darkest red stubborn problems.

Step 3: Solver Mapping – Decontextualizing and Recontextualizing Based on the above-described input, Step 3 focused on a combination of "decontextualizing" and "recontextualizing" two particularly pressing problems (described below). By decontextualizing, we mean stating the fundamental problem in a solution-agnostic way. By recontextualizing, we mean taking that fundamental problem and posing it in such a way that experts from other disciplines will see it as their home problem. Decontextualizing can force a reframing that broadens the solution space. Recontextualizing can target the decontextualized problem to specific outsiders with relevant expertise. These processes can work independently or in combination. In this exercise, we merged the two in order to reach the goal of identifying out-of-discipline experts to target for the summer program.

Before we could begin the process of decontextualizing and recontextualizing, we first had to overcome the initial skepticism about external value. As is typical, our expert participants were very focused on the roadblocks to progress along known dimensions. Even though we had isolated core subproblems in the previous steps, they could not imagine how someone who hadn’t received decades of training and experience in their subdiscipline could move the state-of-the-art forward. To help them overcome these preconceived notions, we structured the discussion as follows:

Step 3a: We started by asking about incumbent stakeholders and known barriers to entry. This allowed participants to verbalize their grievances and have us acknowledge the basis for the skepticism.

Step 3b: Next, we asked the group to explain why they were pursuing the two selected problems – shape and composition – in this way. We asked if anyone had seen any wild ideas in the literature or had ever drawn on insights from other domains using related tools. Where step 3a primarily served to air grievances with the top-down desire for OI, step 3b began to elicit areas for potential follow-up.

Step 3c: Based on the ideas from step 3b, we began to experiment with the grouping of identified subproblems – either further decomposing or integrating into higher levels of abstraction. For example, we realized that some related fields might analyze data at different levels (e.g., some statistical tools yield more powerful insight when fusing multiple sources of data, while this community tended to specialize in particular measurement paradigms). We returned to Step 3b for each of these new groupings.

Overall, the discussion yielded two main decontextualized problems that could productively be targeted at particular areas of outside expertise. The first problem involved inferring the shape of known bodies. Decontextualize: Planetary defense scientists prioritize measurements of (apparent/absolute) magnitude and orbit when determining impact probabilities, but these parameters are intermediate steps if the goal is to design future deflection missions. Much less work has been done to infer a shape catalog of these bodies. This can be performed by reanalyzing existing data, based on the inputs described in Fig. 33.3. Recontextualize: Participants realized that this is fundamentally a statistical meta-analysis that could leverage the huge quantities of intermediate data that have already been processed, without the need for analysts to know the details of the individual parameter estimates.

The second problem is the dearth of information about the internal characteristics of asteroids. Decontextualize: In space, this type of measurement is only possible with a close flyby and, ideally, in situ measurement. However, given that it is likely unrealistic to visit enough asteroids in the near term to make any meaningful large-scale inference, a radically new approach is required. Through the discussion, the group realized that much of what they require is a core element of modern geology and seismology. Recontextualize: While seismologists are not members of the planetary defense community as currently defined, they certainly have expertise in their own right. They could provide new measurement techniques and instrumentation as well as strategies for generalizing characteristics of composition based on their vast experience on earth.

Finally, although not related to a specific problem, the discussion surfaced medical imagers (specifically, those experienced in CT tomography) as potentially being a source of tools to augment current efforts at image reconstruction. They face similar problems and are much better funded.

33.4 Discussion and Broader Implications

This study began with the premise that many complex systems that may not – in their native state – be "prizeable" (i.e., amenable to solving through OI methods) can become "prizeable" through a particular kind of problem decomposition. Specifically, one decouples parts that are suitable for (1) distant expert search, (2) sampling from the right tail, and (3) force multiplying. In this paper, we set out to develop and demonstrate a method that leverages a facilitated expert workshop to elicit those kinds of prizeable problems. Our research context – the NASA Asteroid Grand Challenge – had previously suffered from the perception that there was no meaningful role for OI methods to play; the physics was such that meaningful contributions were dominated by large, multimillion dollar observatories. However, through our workshop, we both demonstrated that prizeable subproblems exist in this setting and that our method is capable of eliciting them. In this concluding section, we reflect on both the local value of this work to the AGC team and more broadly on the (generalizable) lessons learned through this work.

33.4.1 Value of Approach in the Context of the Asteroid Grand Challenge

As noted earlier, the AGC team had spent nearly 2 years seeking to gain traction on a set of problems where the planetary science community would value "open input." This 2-h workshop was successful in that it made meaningful progress toward that goal.

First, it identified two subproblems that discipline experts agreed could provide a productive avenue for external, non-expert contributions. While "could" is hardly overwhelming support, it represents a significant step forward from "not possible, insulting that you would think it could." We believe that this transition in receptiveness was made possible by the way the workshop was structured. Our exercises focused on an objective decomposition and abstraction that organically yielded potentially prizeable subproblems. The experts were generally receptive to exploring new methodologies since the identification of opportunities came from them. We also emphasized the complementarity of OI: it only served to free up their time to focus on problems that only they could solve, not trying to replace their hard-earned expertise.

Second, the workshop identified three areas with high potential for distant expertise: seismology, medical and industrial imaging, and statistics/risk analysis.

The notion of distant expertise has been previously identified in the literature [13, 14] in terms of the ability for experts from other domains to view old problems anew and provide frame-breaking solutions. However, to date, the identification of these distant experts has always been done retrospectively. Our guided brainstorming allowed the group of experts to predict areas where other disciplines could contribute to isolated subproblems within their own domain. While it remains to be seen whether these predictions will bear fruit, an indirect benefit of the act of making these predictions can already be seen. We observed that the process of thinking through the ways in which external experts can contribute served to morph the "crowd" from a mass of lay contributors to a legitimate alternative source of expert input. From a professional identity perspective, distant experts are much more like an out-of-discipline collaborator than "Joe Average" whose wild idea happened to pay off and upstage your life's work. Not surprisingly, this image of the "crowd" is far less destructive.

33.4.2 Value of the Approach in General

More broadly, this effort reinforced and elaborated upon several key issues in the literature. First, while the notion of decomposition as a path to OI has been discussed in the literature [7], it has always been framed as universally positive. However, while the act of decomposing never hurts, neither does it always help in a meaningful way. The AGC team had been proposing decomposed pieces to be challenged for more than a year prior to this workshop. However, since these were not identified by members of the planetary science community, these pieces were peripheral and contributed minimally to the main detection efforts. As such, they were met with resistance and did not fully yield desired results.

Second, the need to smooth the transition from "problem solver" to "solution seeker" [26] was at the forefront of our experience. The overwhelming sentiment of participants was that attempting to apply OI to this space was never going to work and at the same time insulting. However, we found that by working with the experts to do the decomposition – through the facilitated workshop – we not only got better problems and we also broke down some of the resistance. They helped us identify problems where their expertise could be augmented, which was the goal of AGC in the first place.

Third, and relatedly, we observed an opportunity to bring in the end-user early, as a "problem formulator." In the past, the notion of bringing end-users in early has been thought of in terms of buy-in [33]. While this is important to the success of many initiatives, we also learned that "formulation" is a critical role in decomposing for OI. The best architecture to enable OI may look quite different from what domain experts are used to, and as a result, their cognitive entrenchment [9] may limit their ability to see that path. Thus, finding a good formulator needs to balance a relatively high level of technical expertise (required to think through alternative decompositions to the problem) with the reality that too much experience tended to limit their ability to un-"fixate" [34] from the normal way of doing business. For example, in the workshop, one astrophysicist fairly new to planetary defense community and the relevant kind of detection played a vital role in moving the discussion forward.

Finally, we found that the process of decontextualizing the problem – thinking about similar or relevant domains – also helped to overcome initial skepticism with OI. Decontextualizing served to morph the crowd from a mass of lay contributors to an alternative source of input; this seemed to be less destructive to the professional identity of experts. This step can both improve the quality of the solutions received and also the likelihood of usage. With respect to quality, for any given subproblem, the way you pose the problem can have a strong effect on who chooses to solve it. Distant experts need to recognize themselves in the problem, to realize they are able and/or interested in providing a solution. With respect to use, we heard less resistance to "consulting" with seismologists than we did to "sourcing" to the crowd.

33.4.3 Future Directions

This work has taken an important first step to implementing the call to expand the use of OI methods through better problem decomposition and formulation. Future work should build on this by developing tools to support facilitation of this process. These can focus on quasi- or fully automated screening tools to identify prizeable problems or ways to identify good formulators and the "right" set of experts to bring into the formulation process. In both cases, a deeper understanding of all the kinds of subproblems that are amenable to "open" is important. More work is needed to reliably identify these kinds of problems in advance.

References

1. Office of Science and Technology Policy (2012) Implementation of federal prize authority: progress report, March 2012
2. Office of Science and Technology Policy (2012) Implementation of federal prize authority: progress report in response to the requirements of the America COMPETES reauthorization act of 2010. Office of Science and Technology Policy, Washington, DC
3. Zients J (2010) Memorandum for the heads of executive departments and agencies: guidance on the use of challenges and prizes to promote open government. Office of Management and Budget, 8 March 2010
4. Lindegaard S (2010) The open innovation revolution: essentials, roadblocks, and leadership skills. John Wiley & Sons, Hoboken
5. Terwiesch C, Xu Y (2008) Innovation contests, open innovation, and multiagent problem solving. Manag Sci 54(9):1529–1543
6. Boudreau KJ, Lacetera N, Lakhani KR (2011) Incentives and problem uncertainty in innovation contests: an empirical analysis. Manag Sci 57(5):843–863

7. Lakhani K, Lifshitz-Assaf H, Tushman M (2012) Open innovation and organizational boundaries: the impact of task decomposition and knowledge distribution on the locus of innovation. Social Science Research Network, Rochester, NY, SSRN Scholarly Paper ID 1980118, May 2012
8. Szajnfarber Z, Vrolijk A, Crusan J (2014) Exploring the interaction between open innovation methods and system complexity. Presented at the 4th international engineering systems symposium, Hoboken, New Jersey
9. Dane E (2010) Reconsidering the trade-off between expertise and flexibility: a cognitive entrenchment perspective. Acad Manag Rev 35(4):579–603
10. Afuah A, Tucci CL (2012) Crowdsourcing as a solution to distant search. Acad Manag Rev 37 (3):355–375
11. Hinds PJ (1999) The curse of expertise: the effects of expertise and debiasing methods on prediction of novice performance. J Exp Psychol Appl 5(2):205–221
12. Poetz MK, Schreier M (2012) The value of crowdsourcing: can users really compete with professionals in generating new product ideas? J Prod Innov Manag 29(2):245–256
13. Jeppesen LB, Lakhani KR (2010) Marginality and problem-solving effectiveness in broadcast search. Organ Sci 21(5):1016–1033
14. Acar OA, van den Ende J (2016) Knowledge distance, cognitive-search processes, and creativity the making of winning solutions in science contests. Psychol Sci. doi:10.1177/0956797616634665
15. Collins HM, Evans R (2002) The third wave of science studies studies of expertise and experience. Soc Stud Sci 32(2):235–296
16. Rabe CB (2006) The innovation killer: how what we know limits what we can imagine- and what smart companies are doing about it. In: AMACOM Div American Mgmt Assn
17. NASA Centennial Challenges (2009) Astronaut glove challenge. NASA, November [Online]. Available http://www.nasa.gov/directorates/spacetech/centennial_challenges/astronaut_glove/index.html. Accessed 20 Aug 2016
18. Simon H (1962) The architecture of complexity. Proc Am Philos Soc 106(6):467–482
19. Nadler D, Tushman M, Nadler MB (1997) Competing by design: the power of organizational architecture. Oxford University Press, New York
20. Baldwin CY, Clark KB, EBSCOhost (2000) Design rules vol. 1: the power of modularity. MIT Press, Cambridge, MA
21. Galbraith JR (1977) Organization design. Addison-Wesley Pub. Co., Reading
22. Colfer L, Baldwin CY (2010) The mirroring hypothesis: theory, evidence and exceptions, Harvard Business School Working Paper, vol 10, no. 58
23. Ethiraj SK, Levinthal D (2004) Modularity and innovation in complex systems. Manag Sci 50 (2):159–173
24. Johnston C (2012) Netflix never used its $1 Million Algorithm due to engineering costs. WIRED, 16 April
25. Amatriain X (2012) Netflix recommendations: beyond the 5 stars (Part 1). The Netflix Tech Blog, 06 April
26. Lifshitz-Assaf H (2016) Dismantling knowledge boundaries at NASA: from problem solvers to solution seekers. Social Science Research Network, Rochester, SSRN Scholarly Paper ID 2431717
27. NASA (2007) Near-earth object survey and deflection analysis of alternatives – report to congress
28. Kessler J (2015) Additional expert session post Ames Asteroid Workshop, 13 June
29. Galache JL, Beeson CL, McLeod KK, Elvis M (2015) The need for speed in near-earth asteroid characterization. Planet Space Sci 111:155–166
30. JPL NEO Program (2013) Introduction: NASA Near-Earth Object Search Program. Introduction: NASA Near-Earth Object Search Program [Online]. Available: http://neo.jpl.nasa.gov/programs/intro.html. Accessed 24 Oct 2016

31. Spahr T (2014) The Minor Planet Center and the International Asteroid Warning Network, May
32. Asteroid Grand Challenge (2015) Asteroid data hunter challenge. NASA, 12 March. [Online]. Available: http://www.nasa.gov/content/asteroid-data-hunter-challenge-0. Accessed 26 Oct 2016
33. Ancona DG, Kochan TA, Scully M, Maanen JV, Westney DE (2004) Managing for the future: organizational behavior and processes, 3rd edn. Cengage Learning, Mason
34. Jansson DG, Smith SM (1991) Design fixation. Des Stud 12(1):3–11

Chapter 34
A Framework for Measuring the "Fit" Between Product and Organizational Architectures

Zoe Szajnfarber and Erica Gralla

Abstract This paper develops a framework for measuring the "fit" between product and organizational architectures. The ability to measure "fit" in an objective and systematic way is a necessary precursor to understanding the nature of the hidden costs associated with design support tools that are becoming commonplace enablers of complex system design. Specifically, these tools are enabled by significant upfront decomposition – the problem is a priori broken up into a set of loosely coupled tasks that can be worked in parallel, with interactions across tasks routinized and often encoded in computational tools. When this imposed structure fits the problem well, it can drastically speed up design cycles and enable intraorganizational collaboration, by hiding extraneous information and freeing up experts' time to focus on the hardest parts. However, even minor mismatches between the organizational decomposition (people and tasks) and product decomposition (the problem being solved) can cause designers to miss important trades and make poor choices. The proposed measurement framework builds on existing measures from the organizational design literature and systems engineering literature. Our contribution lies in unifying the level of analysis of the two disciplines and developing a novel strategy for tracking the interaction among the product and organizational system. The utility of this approach for observing influences in real systems is demonstrated with a "toy" case study example based on space system development at the Jet Propulsion Laboratory.

Keywords Architecture • Decomposition • Organization • Design structure matrix • Mirroring

Zoe Szajnfarber and Erica Gralla contributed equally to this work.

Z. Szajnfarber (✉)
Engineering Management and Systems Engineering and Elliott School of International Affairs, the George Washington University, Washington, DC, USA
e-mail: zszajnfa@gwu.edu

E. Gralla (✉)
Engineering Management and Systems Engineering, the George Washington University, Washington, DC, USA
e-mail: egralla@gwu.edu

A.M. Madni et al. (eds.), *Disciplinary Convergence in Systems Engineering Research*, https://doi.org/10.1007/978-3-319-62217-0_34

34.1 Introduction

As today's engineered systems become increasingly complex, effective design requires bringing diverse expertise and knowledge to bear on each design iteration, in an expedient way. Faster design cycles and better cross-disciplinary integration are enabled by the ever-increasing power of model-based approaches to concurrent design (e.g., NASA's Team X, EADS Astrium Satellite Design Office). However, these advantages come at a cost, and that cost has not received enough attention. Specifically, concurrent design and engineering is enabled by significant upfront decomposition – the problem is a priori broken up into a set of loosely coupled tasks that can be worked in parallel, with interactions across tasks routinized and often encoded in computational tools. When this imposed structure fits the problem well, it can drastically speed up design cycles by hiding extraneous information and freeing up experts' time to focus on the hardest parts. However, even minor mismatches between the organizational decomposition (people and tasks) and product decomposition (the problem being solved) can cause designers to miss important trades and make poor choices [1, 2]. As these design tools become more popular, it is important to understand their scope of applicability and potential costs.

A critical precursor to understanding the impact of lack of "fit" on the design process is an ability to empirically observe and measure the "fit" between the imposed organizational architecture and the natural decomposition of the technical system. In this paper, we take that first step, carefully defining what is meant by "fit," identifying its key dimensions, and proposing a framework for how it can be measured. The concept of "fit" has previously been explored in the management literature (as in the mirroring hypothesis; [3]) and in software engineering (as Conway's Law); however, we take the notion of matching much farther, tracking the nature of organizational-product interactions that drive goodness of fit. We illustrate (a) the need to adopt consistent levels in characterizing the architectures of the organization and the product and (b) the challenges in observing and tracking the interactions between them through application to an empirical setting.

34.2 Related Literature

Decomposition of complex systems has been studied extensively from a variety of perspectives. The most relevant streams are those that focus on the relationship between the decomposition of a technical product and the decomposition of the organization that designs or produces it.

Product Decomposition Technical products are decomposed because they are complex. Complexity is often managed through task and product decomposition [4, 5]. Specifically, since no individual can practically process the amount of specialized knowledge required to design a complex integrated system, decomposition serves to break the problem into a set of (nearly) decoupled, individually tractable modules [6–8]. Intelligently selected modules are tightly coupled internally and loosely coupled to other modules in the system. This allows work on individual modules to proceed independently and in parallel [8–10].

Organizational Decomposition. Organizations employ decomposition for similar reasons: to manage complexity. If an individual is faced with more information than (s)he can feasibly process, (s)he is no less able to make a decision than if (s)he had too little. Tushman and Nadler [11] view an organization as an information processing system. Hierarchical structures in organizations serve to compartmentalize where decisions need to be made. In these structures, information is "hidden" inside business units, and only some information is flowed up the organization, to limit the scope of information each unit must deal with.

"Fit" Between Product and Organizational Decompositions The literature also addresses the relationship between product and organizational decompositions. The core finding of this literature is the so-called mirroring hypothesis, which states that: the structure of the organization that develops the technology (defined in terms of, e.g., communication links, collocation, team, and firm co-membership) will match the product architecture of the system under development [3, 6, 8]. Substantial effort has gone into demonstrating the empirical validity of the hypothesis [12]. An implication of the hypothesis is that successful design can occur when the organization is less decomposed than the product, but not the other way around.

Level of Decomposition. All technical systems can be decomposed to some degree at relatively low cost. For a given system, the precise level depends on the inherent structure of the system [6, 13]. Beyond this point, systems can be actively decomposed into progressively smaller subproblems by imposing design rules [8], global explicit rules about how a system will operate. The idea is to define key design parameters and guarantee that they will not change; this allows modules to depend on design rules and not on each other. Design rules can take the form of standards (e.g., 802.11b) or be embodied in interface control documents.

Impact of Decomposition on the Design Process While decomposition is necessary, and ensures tractability of the design process, it also constrains the trajectory of the design process in several ways. Because work happens within clearly defined module boundaries, new insights are developed inside particular modules rather than across them [14, 15]. In addition, the decomposition defines how coordination happens across interdependent tasks [4, 11, 16]. Information is "hidden" within modules and shared across them only when the need is clear, which may result in missed opportunities for design trades. For all these reasons, the decomposition influences the design process and the space of possible design solutions that can be explored. When chosen well, decompositions can streamline the design process substantially.

Costs of Over-Decomposing However, product and organizational decompositions are not always chosen well, and the costs of poor decomposition choices are not well understood. While volumes have been written about the value of modular decomposition [8, 10, 17], its costs have only been discussed in a superficial way. For example, costs of structural overhead associated with splitting were discussed by Baldwin and Clark [8]. Ethiraj and Levinthal [1] find that under-decomposing leads to limited search and suboptimal designs, while over-decomposing leads to stalled improvement in design performance, but this is based on a simplified model rather than empirical studies.

The cost of getting a decomposition "wrong," in the sense that the product decomposition does not "fit" the organizational decomposition, has not been examined explicitly, nor has this concept of "fit" been well articulated. Our survey of the extant literature makes clear that the selection of a decomposition strongly influences the trajectory of the design process, so it is important to understand the potential problems that could result: the costs of decomposition.

34.3 Approach

This paper describes an initial attempt to develop a framework for measuring the fit between organizational and product architecture. We began by surveying the literature to identify how each of the organizational design and product architectures is represented in their respective literatures. We then assessed their mutual consistency – could representations of organizational design be contrasted directly with representations of product architectures? We built on concepts in both literatures to develop a framework that enables clear description of the key elements of both product and organizational architectures in the same "language," so that the representations could be directly compared.

We then applied these measures to an example from the domain of space system design to both identify areas where additional resolution is required and also to illustrate the kinds of insights that can be gained. This paper represents a first step toward empirical investigation of the effects of mismatches between organizational and product architectures. The framework will guide empirical data collection and support the analysis of empirical results.

34.4 Diagnosing "Fit" Between Organizational and Product Architecture

We next develop a screening tool to assess the quality of "fit" between an organization and its product architecture. While there are established frameworks for characterizing and representing both the "design" of an organization and the architecture of its product system, there is currently no common basis upon which to measure "fit." We build on existing approaches to develop an appropriate framework for (1) representing (i) the product architecture and (ii) the organizational decomposition, and (2) assessing the "fit" between them and screening for potential issues. Each of these endeavors is described below. As a preliminary test of the screening tool, we apply it to characterize a spacecraft development team.

34.4.1 Representing Architecture

A common way of representing the structure of a system is in a Design Structure Matrix (DSM) [18, 19]. A DSM is an $n \times n$ matrix representation that lists variables on both rows and columns; when an "X" appears in the matrix, it signifies that the variable in its row requires an input from the variable in its column. A DSM is often analyzed and rearranged in order to group together tightly coupled sets of variables into modules. This is useful because these interconnected variables need to be designed simultaneously, usually by an individual or team, while work on different modules can proceed in parallel. The basic DSM structure has been used to document interdependencies among product components or design tasks [19]. We will use the basic DSM structure as a starting point to represent the architectures of both the product being designed and the organization carrying out the design.

34.4.1.1 Product DSM

One DSM will represent the design problem as a product DSM. Product DSMs are fairly standard and aim to capture the interdependencies among modules of the technical system. These DSMs must represent two components. (1-p) The elements of the DSM represent the standard product subsystems (in a spacecraft, e.g., subsystems would include thermal, configuration, power, etc.). (2-p) Many of the subsystems are interdependent (e.g., if the collecting area of the instrument is enlarged, there will be more data to process and downlink and the electronics will require additional thermal regulation). These interdependencies are abstracted as "Xs" in the off-diagonal cells of the DSM.

In the product DSM, more advanced representations also seek to distinguish between local and global dependencies among subsystems. Here we specify two special cases and discuss how they can be represented in a DSM. Traditional interface control documents (ICDs) capture the predefined dependency between two subsystems. For example, the definition of a VGA port guarantees that any monitor can be interfaced with any personal computer or laptop. It allows design decisions internal to each subsystem (i.e., those not affecting the interface) to be made completely independently. Design rules, as coined by Baldwin and Clark [8], explicitly define global dependencies that can be assumed for all subsystems. For example, North American power outlets provide power at 60 Hz. The key difference between these two types of interdependencies lies the ability to change them. While ICDs are intended to be fixed for the life of a project, if there's a compelling

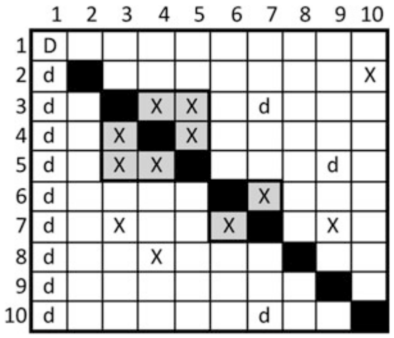


Fig. 34.1 Notional pDSM

	1	2	3	4	5	6	7	8	9	10
1	D									
2	d	■								X
3	d		■	X	X		d			
4	d		X	■	X					
5	d		X	X	■				d	
6	d					■	X			
7	d		X			X	■		X	
8	d			X				■		
9	d								■	
10	d						d			■

reason to modify the standard, a change only needs to be agreed on by the two subsystems affected. If, on the other hand, a change needs to be made that affects a design rule, the effects would ripple through the whole system (and any past systems that relied on that design rule).

In the DSM, the ICD-type dependency (3-p) is represented by replacing the “X” with a “d” to indicate that the dependency has been fixed. A design rule-type dependency (4-p) shows up as a separate element at the top left corner of the DSM. Since it affects most of the other elements of the system, one would expect to see a “d” in nearly all rows of that column. Figure 34.1 illustrates a notional product DSM.

34.4.1.2 Organizational DSM

The second (separate) DSM will represent the structure of the organization doing the work. While product-level task-based DSMs are fairly common and have been used extensively to analyze project attributes like efficient work distribution and ordering (e.g., [18]), the DSM construct has not been used to analyze “standing organizations” (i.e., those designed to work on multiple distinct products that aren’t known a priori) in any detail. This is the view of the organization taken in this research.

For our purposes, the organizational DSM needs to include four critical components.

(1-o) Similar to the pDSM, the elements of the oDSM represent the business units or task owners in the organization. These tend to map to subsystems in the physical system, but that is not always the case. For example, you would likely have a propulsion team that maps to the propulsion subsystem, but you might also have a quality assurance team that does not map directly to a single subsystem. Where in the pDSM, the important distinction is among implicit (2-p), local (3-p) and global (4-p) dependencies, in the oDSM, the important distinction is between passive (routinized) and actively managed (human) interdependencies.

(2-o) In the oDSM, the routinized aspects of organizational interdependencies are captured with “Rs.” They can show up as off-diagonal “Rs,” but more often they mirror the global design rule construct, with one subsystem automatically depending on another through a central (often artificial) layer. As an example of this kind of dependency, many organizations implement an IT backbone that automatically updates each business unit when decisions or commitments made in one unit impact another. For example, in situations where multiple groups draw from a common inventory, when one draws that inventory down, it affects what’s available to the other unit. It is important to note a distinction between the “R” in the oDSM and the “D” in the pDSM: whereas a design rule “D” indicates a static value, so that all subsystems can proceed with their own designs without worrying that the interface with “D” will change, a routinized interdependency “R” indicates that the existence of the *relationship* is static, not necessarily the value passed within that relationship. Continuing the inventory example above, the “R” dependency does not mean that business unit A can rely on a static inventory level of, say, 4. It means that they can check their current allocation through the IT backbone without talking to business unit B, even if their level depends on B’s use.

(3-o) In the oDSM, we also need to capture the common case where interdependencies are handled and negotiated in real time, i.e., actively managed by humans. These are represented as off-diagonal “Hs.” For example, many organizations use cross-disciplinary high-performance teams to dynamically reallocate resources across business units and tasks. It can also happen less formally between two business units. For example, on a technical project, detailed design may have revealed that an assumed process won’t be feasible in practice. This may affect requirements allocated to multiple subsystems, and necessitate a reallocation of that resource.

(4-o) The last aspect of the organization that needs to be captured is the physical layout of the organization or workspace, because it impacts the ease of coordinating across interdependencies (Allen 1997). This involves considering the collocation of particular aspects of work (e.g., in a multinational corporation, which subunits share a facility in a particular location). These constraints can be represented in the DSM as a fixed ordering of certain rows/columns, which means that there are limited options for reordering the DSM (a core aspect of traditional DSM analysis).

Figure 34.2 illustrates a notional oDSM. It takes the same general form as a pDSM, but as noted above, some of the elements take on different meanings. For a given product-organizational pair, the DSMs are often not the same size, since there may be extra organizational units or single organizational units that are responsible for more than one technical subsystem. Rows/columns may have a fixed ordering, represented by the brackets to the right of the matrix.

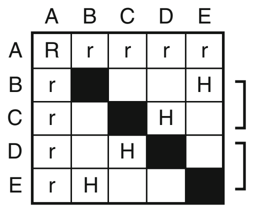


Fig. 34.2 Notional oDSM

34.4.2 Assessing Fit

The next step is to measure the “fit” between the pDSM and the oDSM. As noted above, previous attempts have been coarse and qualitative. For example, MacCormack et al. [3] compare the software architectures of two similar products, one developed by an open source team and the other by a traditional “closed” organization. The study found that “fit” was important because the traditional organization generated a product with a large interconnected core, while the open team produced a product architecture with a comparatively small core and many loosely coupled modules. Since our goal is to assess the impact of lack of fit, we need a more granular and systematic representation.

34.4.2.1 Conceptual Basis: Problems to Be Identified

Before discussing the mechanics of how to measure “fit,” it is important to be clear about the conceptual intent of the proposed framework. It needs to be able to identify potential problems (which can then be further investigated) resulting from a lack of “fit” between the pDSM and the oDSM. Extant theory on fit provides some guidance on the types of problems we expect to find. In the following paragraphs, we use this theory to develop a list of the types of problems the framework must identify.

At the highest level, a good “fit” is defined as a perfect overlap between (a) the elements in each DSM and (b) their respective feedback structure. Mismatches occur when a technical element or feedback does not map to an organizational counterpart or vice versa. While it is more problematic for a need for technical feedback to exist where there is no organizational system to facilitate it, misplaced institutional structures (e.g., a cross-functional team where no tradeoff needs to be made) represent a wasteful overdesign. Therefore, the framework needs to be able to directly compare the pDSM and oDSM in terms of their joint ability to handle necessary feedback in the design process.

A structural mismatch does not necessarily indicate a problem. Evaluating the potential impact of that mismatch requires some understanding of its potential outcome. Prior research has highlighted several kinds of impacts that relate to mismatches.

First, it is known that routinizing information flow can have substantial advantages in terms of process efficiency [2]. Since everything happens automatically, the process tends to fade into the background and is unobservable except in terms of the process artifact. This is great when the routinized structures fit the natural communication flows of the product; however, even slight mismatches can cause problems if the organization forgets that hidden processes were enabling smooth operations, and fails to evolve [2]. An unaccounted-for divergence between the product (which changes from, e.g., one project to the next) and the organization (which stays the same) can lead to two types of problems in our framework:

- Problem type 1: Lack of institutional pathway where one is needed. Diagnosing this issue requires two levels of analysis, elaborated upon below. First, in terms of structural match between the pDSM and oDSM, this would involve an off-diagonal X in the pDSM, matched by only an “r” in the oDSM or potentially no structure at all. Second, the impact can be observed in terms of the “extra” communication required to deal with the technical feedback in the unplanned organizational location. This might be observed as pulling leads from several subsystems into an emergency meeting to resolve an issue discovered late in the process.
- Problem type 2: Overdesign (excess institutional pathways). Now instead of the change in product leading to a lack of institutional pathway where one is needed, an overdesign occurs where a legacy communication paths remains where it is no longer needed. In the structural sense, this would show up as an oDSM “h” with no counterpart in the pDSM. In terms of communication, one would observe little or none in a location where a lot is expected (e.g., a standing cross-functional team exists).

The product analog to the organizational routinization of information flow is embodied in the concept of a predefined design rule. An important intent of a product decomposition is to enable subtasks to be executed independently and/or in parallel [6, 8]. This can add substantial process efficiencies, as long as the decomposition is clean. Ethiraj and Levinthal [1] have shown that a poor decomposition – one where the divisions do not match the natural structure of the system – can do more harm than good. Technical issues don’t surface until late in the design process or inefficient designs are chosen so that everyone can fit in their poorly allocated requirements.

- Problem type 3: Poor decomposition (not accounting for a technical interdependency). In this case, no structural mismatch between the pDSM and oDSM would surface. The issue is that a technical feedback was not represented in the pDSM when it needs to be. To identify a problem type 3 requires observation of the in-process communications. Here, one would observe high incidence of communication where none is expected (i.e., the pDSM is empty and the oDSM is likely empty too (though that is not required)).

In considering levels of communication, it is important to recognize that not all incidences of high communication are indicative of a problem. An important systems engineering function involves brokering in a structured way across predefined interfaces. This is particularly important early in the design process. Therefore, if a high level of communication is observed – especially if it happens early – in a location where there is both a product feedback and an organizational feedback, it is simply evidence of the organization working as it should. Our framework enables the identification of these problems.

34.4.2.2 Generating a Structural Matching Matrix (mDSM)

The first step in assessing fit involves a direct comparison of the structural similarities of pDSM and oDSM. To do this first requires that the pDSM and the oDSM be matched in terms of size – accounting for the fact that some organizational elements don't have technical counterparts and that some organizational elements deal with more than one technical subsystem. For the notional example above, this might look like Fig. 34.3. Business units B and C are responsible for three technical subsystems each, and business unit D is responsible for two. The assignment of pDSM elements to oDSM elements should follow the actual assignment of work in the organization. If any pDSM element has no mapping to the oDSM, it can be added as a row/column in the oDSM; this would be a serious oversight on the organization's part and therefore is likely a rare occurrence in real organizations.

In order to create a matrix like Fig. 34.3, a second transformation may also be required. Recall that the order of the oDSM rows/columns is often fixed, to represent the collocation of some teams or personnel. Therefore, the pDSM may require reordering in order to map to the oDSM structure. In our example, we assume that the pDSM has already been reordered so that pDSM elements 2, 3, and 4 are all assigned to business unit B, etc. The difference between an optimized pDSM (e.g., one in which above-diagonal interfaces are minimized) and the pDSM required to match the oDSM is one measure of the cost of a decomposition. This will be explored in future work.

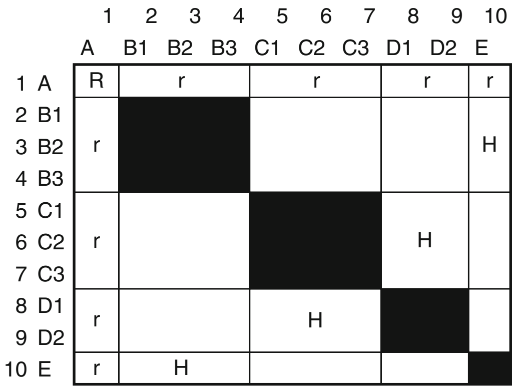


Fig. 34.3 Transformed oDSM

Fig. 34.4 mDSM

		1	2	3	4	5	6	7	8	9	10
		A	B1	B2	B3	C1	C2	C3	D1	D2	E
1	A	0	1	1	1	1	1	1	1	1	1
2	B1	0	0	1	1						0
3	B2	0	1	0	0	-1		-1			1
4	B3	0	1	0	0	-1					1
5	C1	0		-1	-1	0	1	1	1	0	
6	C2	0				1	0	0	1	1	
7	C3	0		-1		1	0	0	1	0	
8	D1	0			-1	1	1	1	0	1	
9	D2	0				1	1	1	1	0	
10	E	0	1	1	1			-1			0

With this transposition, the reordered pDSM can be "subtracted" from the modified oDSM. We will record the difference (oDSM-pDSM) as follows. Our example mDSM is shown in Fig. 34.4.

- 0 = an element or feedback was present in both structures.
- Blank cell = no elements were present in either structure.
- 1 = an element or feedback was present in the oDSM but not the pDSM.
- −1 = an element or feedback was present in the pDSM but not the oDSM.

34.4.2.3 Screening for Issues Using Information Flow (iDSM)

A mismatch among the pDSM and oDSM becomes an issue when communication that should have happened does not happen, and an inefficient design and/or design process results. Alternatively, an organization can be "overdesigned" to accommodate information flows that never materialize. Section 34.4.2.1 identified specific types of problems that might occur. In terms of the framework, these situations can be operationalized as follows:

- Problem type 1: Lack of institutional pathway where one is needed = −1 AND high level of information flow
- Problem type 2: Overdesign (excess institutional pathways) = 1 AND limited information flow
- Problem type 3: Poor decomposition = blank cell AND high level of information flow
- Normal process = 0 AND information flow

Clearly, a measure of information flow is needed to assess whether the mDSM indicates that there are problems. We propose the use of a parallel matrix, termed an iDSM, to track the intensity of information flow actually required during the design process. The iDSM distinguishes intense from limited information flow in the

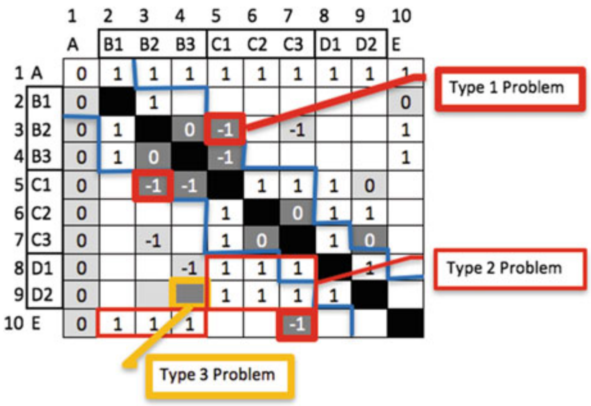


Fig. 34.5 iDSM overlaid on mDSM

interaction among the matrix elements. If we superimpose the iDSM over the mDSM, we can distinguish the problematic interfaces from those that are appropriately managed. Figure 34.5 provides an example. The iDSM provides the color for the cells, while the value of the cell is taken from the mDSM. Cells in light gray exhibit limited information flow, and cells in dark gray exhibit intense information flow.

Problems can now be identified. Most of the dark gray cells are not problematic, because they are appropriately managed by grouping tasks within business units or by actively managed interfaces. However, there are several dark gray cells that contain −1 entries. These indicate a lack of institutional pathways where one is needed (problem type 1). There is also a dark gray cell that is blank, which indicates a poor decomposition (problem type 3): the interdependency of these elements was not recognized in the pDSM. Another set of problems is indicated by the white cells that contain 1 entries: these may indicate overdesign (problem type 2), in which institutional pathways exist where they are not needed. In most cases, these are the result of organizational units managing more than one technical element, but not in all cases; these latter are likely more problematic.

A second layer should be considered in assessing these results. Recall that the ordering of the rows/columns indicates collocation, which eases communication. If two teams are collocated, they may be able to create an actively managed interface even if it was not planned for originally. The blue lines in Fig B outline a notional border within which such "on the fly" active management is feasible: they include entire business units (such as B1-B3) and rows/columns that are directly adjacent (such as B3 and C1) on the assumption that these subteams are next to one another. Under these conditions, a few additional problematic interfaces (C1-B3) may be appropriately managed.

34.5 Preliminary Validation of the Approach

For any screening tool to be useful, it must be implementable in a real-world context. This generally means that (1) the constructs both have meaning in and fully describe empirical settings and that (2) the required data is available or at least collectable. Therefore, in this section, we first consider data needs for each of the base DSMs (p, o, and i) and then apply the constructs in a very preliminary way to a setting we hope to study further in the future.

34.5.1 Data Inputs

The information needed to produce the pDSM is generally well documented in most organizations. It can therefore be constructed based on source documents such as blueprints, architecture drawings, or more generally design documentation. For example, Suh et al. did this for Xerox machines [20].

The same is generally true for oDSMs. The bulk of the information needed to construct an oDSM will be stored in documents like organization charts that define business unit leads or cognizant engineers (CogEs) or similar. Since the standing up of high-performance teams or the use of liaisons between groups can be somewhat ephemeral, it may be harder to find explicit documentation.

Where the pDSM and oDSM can be reconstructed from archival documents, the iDSM is more difficult because information flow is not generally captured in such documents. Instead, information flow is revealed by the meetings, emails, and other communication that occurs over the course of the project. There is a lot of communication that is internal to a business unit (e.g., unit B may lead subsystems 3–5 which strongly influence each other). To limit the scope of required data collection, we will intentionally focus on across-unit communication, because this is where mismatches tend to be most relevant.

Since this type of data is rarely documented, it must be gathered by observing the progression of a project. The specific content of the information will be project dependent – a large scale project will have a sequence of formal reviews and associated structured technical interchanges, where a conceptual design effort may be limited to a few team “huddles.” Level of information flow is therefore a relative concept. For our present purposes, a simple categorization of information flow as “high” or “low” is sufficient. In the future, it may be important to track information flow over time. As noted above, high levels of flow late in the process can be important. It will also be interesting to record when information flow actually impacts the design. These issues will be recorded but not used at this stage of the framework development.

34.5.2 A Preliminary Test of the Tool

As an initial test of this framework, we have identified a suitable empirical setting to exercise it. JPL's Team X is widely considered an exemplary concurrent engineering team, and they have completed more than a thousand studies since 1994 [21]. Their success led to requests to use their services for a wider variety of design problems, including orbiters, landers, and rovers. While the mission type varies, the fundamental structure of Team X changes very little from study to study. As a result, we can observe the design process across studies with varying fit between the organizational and product architectures.

Team X consists of a set of subject matter experts (SMEs) covering the main subsystems required for spacecraft design, and a set of tools and facilities to support their work. The main facility is a room with workstations for each subsystem; each workstation has a set of Excel spreadsheets that capture the evolving design of the subsystem, and each sheet includes links that push and pull parameters from other subsystems. The spreadsheets function as models of the subsystem, and include assumptions about the relationships among parameters. For example, a spreadsheet might include an assumption that the mass of the spacecraft bus is proportional by some factor to the mass of the payload. The spreadsheets are designed by the subject matter experts based on best practices in subsystem design and data from past missions. Team X typically conducts three-day design studies that end with a feasible "point design" for a spacecraft. In other words, the team begins with a set of customer requirements (such as a mass limit, cost cap, pointing requirements, etc.) and a basic architecture (such as an orbiter or a lander), and designs a spacecraft that meets these requirements.

Going forward we intend to use the screening tool to study the impact of different levels of mismatches between Team X's organizational and product architectures. In this initial study, we have a more modest goal. Table below uses the Team X context to verify that our tool's constructs have meaning and describe the setting in a useful way.

From the table we can already see how the tool will allow us to quickly hone in on potential problem areas that merit additional follow-up. For example, the Team X facilitator plays an explicit brokering role to resolve issues across subsystems. Therefore, we need to be able to distinguish high communication led by a facilitator from a similarly high level of communication led by a technical SME. In the first case, the high communication is normal whereas the second case is indicative of a problem wherein a technical SME might be forced to fill the same brokering function due to a mismatch. Layering the iDSM constructs on the mDSM (oDSM-pDSM) will let distinguish these cases as intended.

In future work, we intend to refine the framework through observation of a pilot study, then use it to examine the issues discussed in this paper.

DSM	Construct	Generic meaning	Examples from application to team X	Information source
pDSM	Rows & columns	Elements of product	Technical spacecraft subsystems (e.g., power, thermal, data handling)	Design documents
	Off-diagonal "X"	Physical interdependency, not governed by pre-established rule	Dependency between two subsystems (e.g., larger collecting area drives larger downlink needs)	Design documents and spreadsheet assumptions
	Off-diagonal "d"	Defined interface between two subsystems	Parameter value fixed between two subsystems (e.g., instrument agrees to s/c connector)	Design documents
	"D" as a module	Global design rule fixed for relevant subsystems	Parameter value fixed for whole system (e.g., 5 V power)	Design documents
oDSM	Rows & columns	Units within the organization	Represented by a "chair" where a technical SME sits (e.g., systems engineer).	Room layout and team staffing
	"R" as a module	Routinized (static) relationships between units or among all units	Technical spreadsheets govern routinized interactions (e.g., when subsystem enters bandwidth, it impacts mass roll-up)	Spreadsheet tool
	Off-diagonal "H"	Actively managed (by a human) dependencies between units	The study facilitator calls scheduled check-ins where system level trades are made. Informal collaborations also crop up.	Planned discussions from schedule. Informal huddles.
	Brackets	Defined geographical spacing of units	Instrument team often sequestered to separate room. Chairs in fixed locations, intended to bring closely collaborating SMEs closer together.	Room layout and team staffing
iDSM	Light gray shading	Low information flow	The deputy systems engineer periodically checks in with SMEs slow to populate the spreadsheet, to get a sense of their progress	Observation of design work
	Dark gray shading	High information flow	The facilitator called an unscheduled meeting between three subsystems when it became clear that the design might not close.	Observation of design work

34.6 Conclusion

In this paper, we set out to develop a framework to assess the “fit” between organizational and product architectures. We built on existing constructs in the literature to generate comparable organizational and product DSMs. Our modest contribution in that context is a formal framework for capturing feedback (in the pDSM) and feedback management (in the oDSM) at a constant level of analysis suitable for cross-comparison. The main contribution is in moving beyond the established notion of structural similarity to consider how structural mismatches actually create problems. By layering observed information flows (iDSM) on the matching matrix (oDSM-pDSM), we are able to quickly identify potential problem areas – where, for example, the institutional feedback mechanism is insufficient to handle necessary product feedback. As a preliminary validation of the utility of the tool, we applied it JPL’s Team X. In future work, we intend to use that setting to further probe the implications of the identified mismatches.

Acknowledgments This work was supported by the National Science Foundation Grant CMMI-1563408.

References

1. Ethiraj S, Levinthal D (2004) Modularity and innovation in complex systems. Manag Sci 50 (2):159–173
2. Henderson RM, Clark KB (1990) Architectural innovation: the reconfiguration of existing product technologies and the failure of established firms. Adm Sci Q 35(1):9–30
3. MacCormack A, Baldwin C, Rusnak J (2012) Exploring the duality between product and organizational architectures: a test of the ‘mirroring’ hypothesis. Res Policy 41(8):1309–1324
4. Thompson JD (1967) Organizations in action: social science bases of administrative theory. McGraw-Hill, New York
5. Williamson OE (1991) Comparative economic organization: the analysis of discrete structural alternatives. Adm Sci Q 36(2):269–296
6. Simon HA (1962) The architecture of complexity. Proc Am Philos Soc 106(6):468–482
7. Nadler D, Tushman M (1997) Competing by design: the power of organizational architecture. Oxford University Press, Oxford
8. Baldwin C, Clark K (2000) Design rules, volume 1: the power of Modularity. MIT Press, Cambridge, MA
9. Morelli MD, Eppinger SD, Gulati RK (1995) Predicting technical communication in product development organizations. IEEE Trans Eng Manag 42(3):215–222
10. Ulrich KT, Eppinger SD (1995) Product design and development. McGraw-Hill
11. Tushman ML, Nadler DA (1978) Information processing as an integrating concept in organizational design. Acad Manag Rev 3(3):613–624
12. Colfer LJ, Baldwin CY (2016) The mirroring hypothesis: theory, evidence and exceptions
13. Szajnfarber Z, Vrolijk A, Crusan J (2014) Exploring the interaction between open innovation methods and system complexity. In: Proceedings of the 4th International Engineering Systems Symposium
14. Parnas DL (1972) On the criteria to be used in decomposing systems into modules. Commun ACM 15(12):1053–1058

15. Prencipe A (1997) Technological competencies and product’s evolutionary dynamics: a case study from the aero-engine industry. Res Policy 25(8):1261–1276
16. Galbraith J (1977) Organization design: an information processing view. Interfaces 4(3):28–36
17. Schilling (2000) Toward a general modular systems theory and its application to interfirm product modularity. Acad Manag Rev 25(2):313–334
18. Steward DV (1981) The design structure system: a method for managing the design of complex systems. IEEE Trans Eng Manag 28(3):71–74
19. Eppinger SD, Browning TR (2012) Design structure matrix methods and applications. MIT Press, Cambridge, MA
20. Suh ES, Furst MR, Mihalyov KJ, de Weck O (2009) Technology infusion for complex systems: A framework and case study. Sys Eng 13(2)
21. Sherwood B, McCleese D (2013) JPL innovation foundry. Acta Astronaut 89:236–247

Chapter 35
Developing an Effective Optical Satellite Communications Architecture

Frank E. Skirlo, Adrien Sullivan, and Abbas K. Saidi

Abstract The recent emergence of optical satellite communications (SATCOM) offers several advantages over traditional radio frequency (RF) SATCOM capabilities. These include achieving higher bandwidths, minimizing the effects of jamming, providing low probabilities of detection and intercept (LPD/LPI), and requiring lower satellite size, weight, and power (SWaP). However, optical SATCOM capabilities have limitations that can make it undependable for certain uses such as in establishing satellite-to-ground links due to the effects of cloud cover. Clouds can completely absorb or refract optical signals and have the most detrimental effects on optical links passing through the Earth's atmosphere. RF SATCOM offers a variety of advantages that optical SATCOM does not, including the ability to operate reliably through the atmosphere and broadcast over large regions, though the RF spectrum is becoming a scarce resource that is increasingly difficult to manage and share among competing users. As a disadvantage, optical satellite links are also difficult to acquire and maintain because of their narrow beam widths. This paper proposes four optical SATCOM architectures that can mitigate the cloud cover problem using Geostationary Earth Orbit (GEO) satellites, along with intermittent Medium Earth Orbit (MEO) or Low Earth Orbit (LEO) satellites, to improve link availability.

Keywords Optical satellite communications • Geostationary Earth Orbit • Medium Earth Orbit • Low Earth Orbit • Colored Petri nets

F.E. Skirlo (✉)
CACI Technologies, Inc, Arlington, VA 22201, USA
e-mail: rank.e.skirlo.ctr@mail.mil

A. Sullivan
National Science Foundation, 4201 Wilson Boulevard, Arlington, VA 22230, USA

A.K. Saidi
Department of System Engineering and Operations Research, George Mason University, Fairfax, VA 22030, USA

A.M. Madni et al. (eds.), *Disciplinary Convergence in Systems Engineering Research*, https://doi.org/10.1007/978-3-319-62217-0_35

Acronyms

ACK	Acknowledgment
B	Blue
CPN	Colored Petri Nets
CPR	Cloud Profiling Radar
ECE	Electrical and Computer Engineering
Gbit	Gigabit
GEO	Geostationary Earth Orbit
HALE	High-Altitude Long-Endurance
LEO	Low Earth Orbit
LOS	Line-of-Sight
LPD	Low Probability of Detection
LPI	Low Probability of Intercept
Mbit	Megabit
MEO	Medium Earth Orbit
ML	Mark-up Language
PCFLOS	Probability of Cloud-Free Line-of-Sight
R	Red
RF	Radio Frequency
SATCOM	Satellite Communications
SWaP	Size, Weight, and Power
TCOM	Telecommunications
UAS	Unmanned Aerial System

35.1 Introduction

35.1.1 Overview and Background

The recent emergence of optical SATCOM offers several advantages over traditional RF SATCOM capabilities. These include achieving higher bandwidths, minimizing the effects of jamming, providing LPD/LPI, and requiring lower SWaP. However, optical SATCOM capabilities have limitations that can make it undependable for certain uses such as in establishing satellite-to-ground links due to the effects of cloud cover. Clouds can completely absorb or refract optical signals and have the most detrimental effects on optical links passing through the Earth's atmosphere. RF SATCOM offers a variety of advantages that optical SATCOM does not, including the ability to operate reliably through the atmosphere and broadcast over large regions, though the RF spectrum is becoming a scarce resource that is increasingly difficult to manage and share among competing users. As a disadvantage, optical satellite links are also difficult to acquire and maintain because of their narrow beam widths. Optical beam widths can measure in millimeters and can travel thousands of kilometers. Various optical SATCOM architectures can be analyzed to help determine how best to mitigate the cloud cover problem using GEO satellites, along with intermittent MEO or LEO satellites, to improve link availability. The method we model these alternative architectures is through a discrete event system called Colored Petri Nets (CPN) [1], developed by Carl Petri at Darmstadt University [2].

35.1.2 Research Goals

Our objective is to determine the best SATCOM architecture to mitigate the cloud cover issue, thereby minimizing packet transmission times and increasing packet transfer efficiency. In order to accomplish this, we would optimally want a 0.99 availability for all uplinks and downlinks along the transmission path. Key operational parameters such as lost packets, lost links, and average packet transmission time will be used as evaluation criteria.

35.1.3 Approach

The analysis and evaluation of alternative "bent pipe" architectures will include a transmitting Earth terminal, a geostationary (GEO) satellite, and a receiving terminal. A second architecture will include two intermittent Low Earth Orbit satellites (one at the transmitting terminal, the other at the receiving terminal). A third will increase the LEO satellites twofold (two at the transiting terminal and two at the receiving terminal). Finally, a fourth will have the same elements as the third, only using Medium Earth Orbit (MEO) satellites in place of the four LEO satellites. The table below provides the four analyzed architectures (Table 35.1).

The figure below provides an operational view of Test Case 4 as a sample optical SATCOM architecture test case (Fig. 35.1).

35.1.4 Results

We conducted CPN simulations for all four test cases. Test Case 4 (SATCOM Architecture with a GEO satellite and multiple LEO satellites) performed best, having the least number of lost packets (14 of 100), and the fastest transmission time to send all 100 packets (16,965 ms). The extreme length of time to send all packets is a result of multiple retransmissions of lost packets due to lost line-of-sight (LOS) due to cloud cover, delays from sending ACKs/NACKs back to sending stations, and the subsequent resending of packets (one-way uplink and downlink delay is 119.3 ms, see Table 35.3 for calculation). Test Case 3 performed second best, followed by Test Case 2, then Test Case 1.

Table 35.1 Alternative SATCOM architectures

Test case	Architecture
Case 1	Earth terminal to GEO
Case 2	Earth terminal – LEO – GEO
Case 3	Earth terminal – Multiple MEOs – GEO
Case 4	Earth terminal – Multiple LEOs – GEO

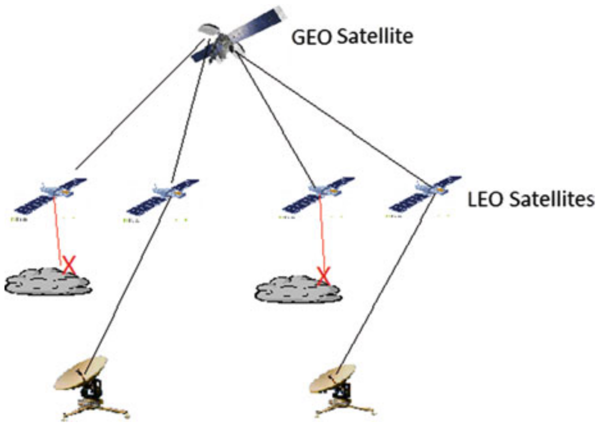


Fig. 35.1 Test Case 4 as an alternative architecture

Table 35.2 CPN input and output variables

Input variables	
Number of packets	100
Types of satellites	LEO, MEO, GEO
Number of satellites (per uplink/downlink)	1, 2
Output variables	
Lost packets	$\leq$15
Maximum time to transmit	$\leq$ view time of satellite

35.1.5 Conclusions

Though we conducted a limited test, our results indicate that cloud cover is a major impediment in developing a viable optical SATCOM architecture. Ways to better mitigate this problem need to be found, such as increasing the number of intermittent LEO satellites using more geographically dispersed ground terminals or operating in areas that have fewer clouds (e.g., desert areas). Unmanned Aerial Systems (UAS) can also be used as pseudo-LEO satellites (pseudolites) and flown in cloud-free areas to maximize LOS opportunities.

Table 35.3 RF link parameters

	Altitude (above sea level) (km)	Latency (t = distance/speed of light (c) (ms)
Earth term – LEO	1000	3.3
Earth term – MEO	10,000	33.3
Earth term – GEO	35,786	119.3
LEO – GEO	34,786	1160.0
MEO – GEO	25,786	86.0

35.2 Experimentation Method

35.2.1 Overview

For all architectures, we transmitted IP packets 1 megabits (Mbit) in length, at a 1 gigabit/sec (1 Gbit/s) data rate (1 ms time slot). Input and output variables (graded against measures of performance) are listed in Table 35.2.

35.2.2 Modeling Probability of Cloud-Free LOS

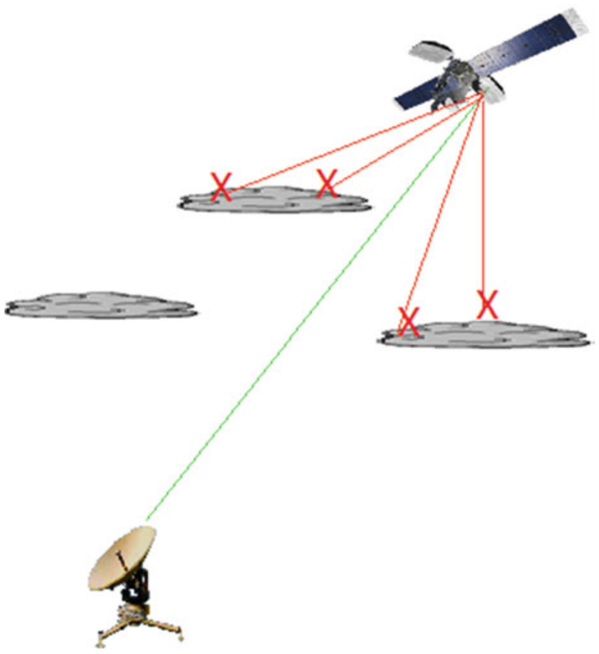
Architectures will be modeled with the effects of cloud cover using recent calculations of Probability of Cloud-Free Line-of-Sight (PCFLOS) based upon arrival/departure angles from nadir [2]. Since clouds tend to maintain a more horizontal orientation, higher angles from nadir tend to have higher PCFLOS then lower angles. Based upon this data obtained from [2], we estimated the PCFLOS as 0.8 for LEO satellites, 0.6 for MEO satellites, and 0.5 for the GEO satellite case (Fig. 35.2).

35.2.3 Other Parameters Included in Architecture Models

Several parameters were included in the four optical SATCOM architecture models inkling packet latencies, view times (from the ground terminal) for LEO and MEO satellites, and latencies for acknowledgements (ACK). The following table provides computed transmission latencies between Earth terminals and satellites, and between satellites (Table 35.3).

To determine satellite view times, one must consider LEO/MEO satellite uplink/downlink look angles (angle from nadir and the Earth terminal) (δ). A sample calculation of the LEO satellite is shown below, where θ is the elevation angle to the satellite from E (Earth terminal), and γ is the internal angle (angle SCE) [2]. The view time calculation is based upon the formula T2 = $(4\pi^2 a^3)/\mu$, where

Fig. 35.2 PCFLOS versus divergence from nadir



$\mu = 3.9860044 \times 105\ \text{km}^3/\text{s}^2$ (Kepler's constant) [4]. A LEO satellite with an altitude of 1000 km has an orbital period of 1 h and 36.4 min. Multiplying this period by $2\gamma/360$ will provide the view time of the satellite from the Earth terminal. Assuming a minimal elevation angle of 10° above the horizon, one can use the Law of Sines to find δ. Since the radius of the Earth is 6378 km, sin(90° + 10°)/7378 km = sin(δ)/6378 km. Solving this equation results in $\delta = 58.32°$ and $\gamma = 180 - 58.32 - 100 = 21.64°$ (Fig. 35.3).

A consolidated table for all three orbits is shown below.

We also included intermittent IP packet receipt ACK along the link path to avoid extreme end-to-end (E2E) delays if a packet was lost early in the path (due to cloud cover or a collision). If packets were not received based upon the latencies in Table 35.4, it is assumed the packet was lost, and another packet was retransmitted from the originating Earth Terminal. We added retransmission latencies backed upon whether the packet was lost on the near-side our far side of the transmitting Earth terminal.

Table 35.5 lists other assumed values for the model and time-out windows along the path.

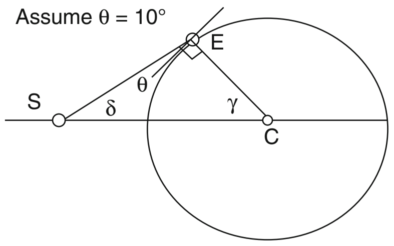


Fig. 35.3 Sample satellite look angle parameters

Table 35.4 Computed satellite: Earth terminal look angle and view times

	Altitude (above sea level) (km)	Maximum look angle (from nadir)	Satellite view time millisecond (min)
Earth term – LEO	1000	58.32°	682,380 (11.4 min)
Earth term – MEO	10,000	23.54°	6,657,699 (111.0 min)
Earth term – GEO	35,786	10° (assumed)	∞ (constant)

Table 35.5 Other assumed parametric values within model

Parameter	Value
Link acquisition delay	90 ms [4]
Time-out window to LEO	682,380 ms
Time-out window to MEO	6,675,699 ms
Time-out window to GEO	Continuous

35.2.4 Test Case 1: Earth Terminal – GEO

This is the easiest of the four cases to model, having only three major nodes (shown as circles), namely sending and receiving nodes and a single GEO satellite. Within CPNs, transitions indicate an event such as sending a packet to the next node (indicated by boxes). Latencies are applied to various transitions as packets proceed through the network. Link availability can also be applied using PCFLOS on uplink and downlink transitions. Both near-side and far-side ACKs are applied to see if sequential packets arrived the satellite of distant ground station. Packets not received at the distant ground station generate a new packet at the sending station. Packets are indicated by tokens (shown in green) which are time-stamped and updated as they pass through the network. Figure 35.4 depicts the CPN for Test Case 1.

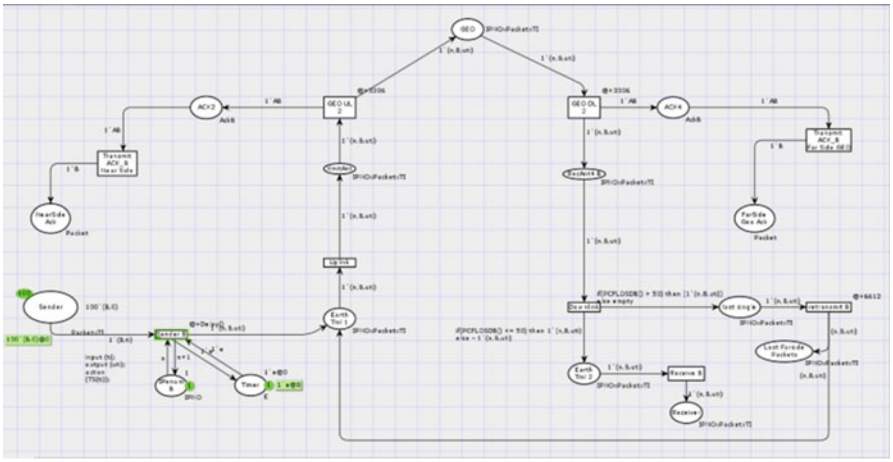


Fig. 35.4 Test Case 1 CPN: Earth terminal – GEO

35.2.5 Test Case 2: Earth Terminal – LEO – GEO

In this case, a single LEO satellite is added to both the near-side and far-side of the network. ACKs are added at the LEO satellites to track if a packet is lost during the uplink or downlink due to cloud cover. There is no need to apply them to the GEO uplink or downlink since LEO satellites are above the clouds. Tokens are shown at various stages of the network, showing their respective network clock times (in milliseconds). Figure 35.5 depicts the CPN for Test Case 2.

35.2.6 Test Case 3: Earth Terminal – Multiple MEOs – GEO

In this case, two MEO satellites are added to both the near-side (uplink) and far-side (downlink) of the network, forming separate, redundant paths. Duplicate packets (tokens) are designated as either red or blue, depending on which path they follow. At the far-end terminal, a request to retransmit a specific packet is sent only if both red and blue packets fail to arrive. Figure 35.6 depicts the CPN for Test Case 3.

35.2.7 Test Case 4: Earth Terminal – Multiple LEOs – GEO

In this case, two LEO satellites are added to both the near-side (uplink) and far-side (downlink) of the network, forming separate, redundant paths. Again, duplicate packets (tokens) are designated as either red or blue, depending on which path they follow. At the far-end terminal, a request to retransmit a specific packet is sent only

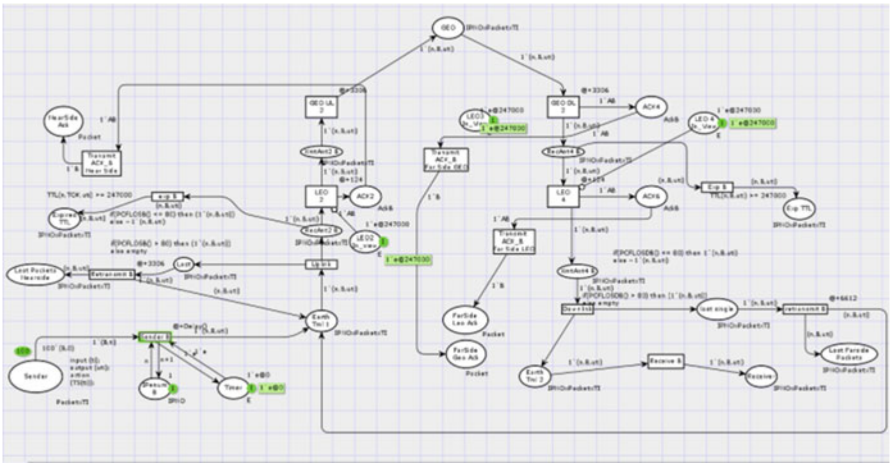


Fig. 35.5 Test Case 2 CPN: Earth terminal – LEO – GEO

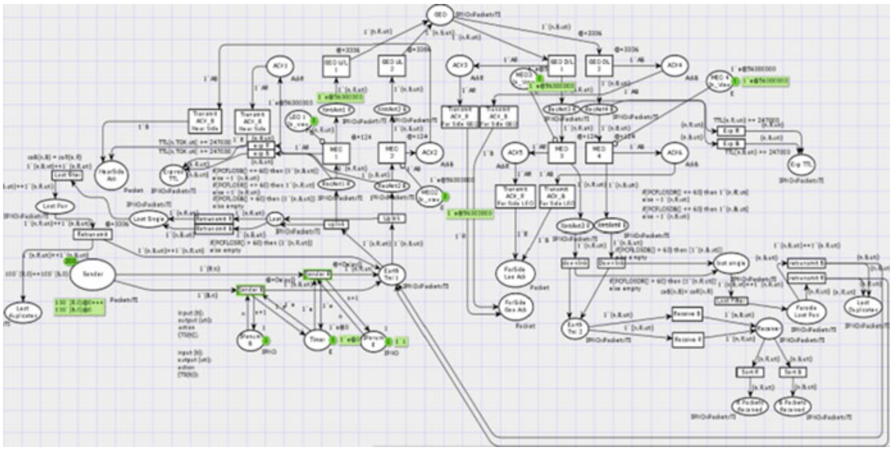


Fig. 35.6 Test Case 2 CPN: Earth terminal – multiple MEOs – GEO

if both red and blue packets fail to arrive. Figure 35.7 depicts the CPN for Test Case 4.

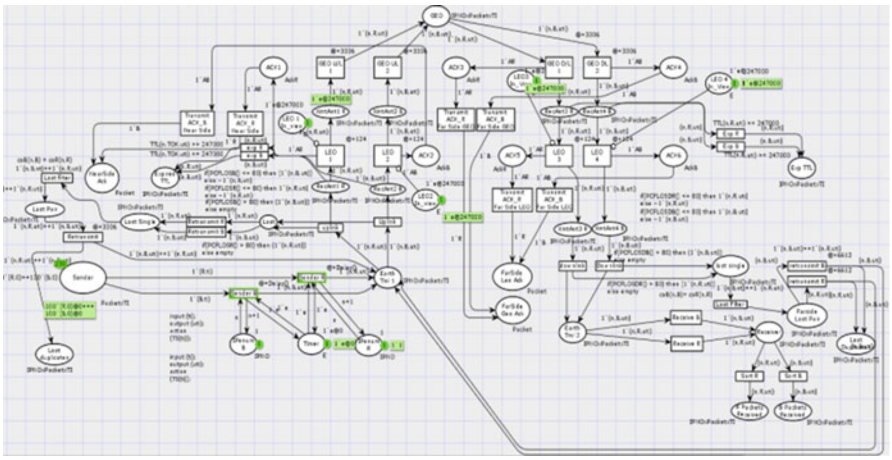


Fig. 35.7 Test Case 2 CPN: Earth terminal – multiple LEOs – GEO

Table 35.6 Results from model simulations

	Total lost packets	Maximum transmission time (milliseconds)
Case 1	91	74,523
Case 2	60	50,604
Case 3	36	29,887
Case 4	14	16,965

35.3 Results

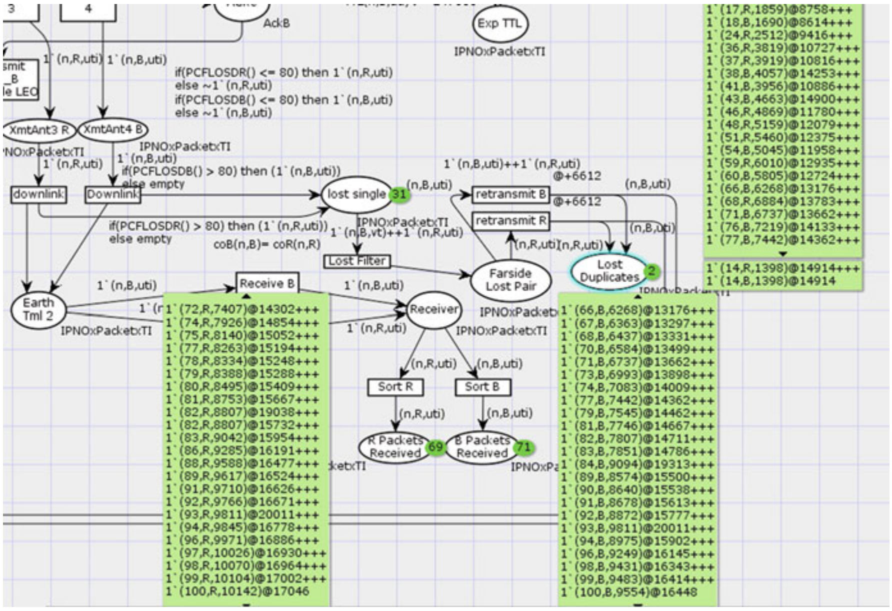
The results of the four test cases are reflected in Table 35.6. Total packet lost totals included retransmitted packets. Having redundant paths provided by intermittent MEO/LEO satellites greatly reduced lost packets and reduced transmission (and retransmission) time for all 100 packets (Test Cases 3 and 4).

The decision support matrix below evaluates the four scenarios in terms of the stated evaluation criteria. From this matrix, Case 4 appears to be the best alternative (Table 35.7).

A sample diagram illustrating the lost packet analysis is shown below. Here one can differentiate between degraded performance with the loss of one data link, or a complete loss of data through the loss of both redundant packet pairs (Test Case 4). Tokens are triplets containing packet number, R/B (for red/blue path), and origination time). The time after the @-sign is the arrival time at the far-side terminal (Fig. 35.8).

Table 35.7 Decision support matrix

	Total lost packets	Average packet transmission time
Case 1	4 (worst)	4 (worst)
Case 2	3	3
Case 3	2	2
Case 4	1 (best)	1 (best)



```
1`(72,R,7407)@14302+++
1`(74,R,7926)@14854+++
1`(75,R,8140)@15052+++
1`(77,R,8263)@15194+++
1`(78,R,8334)@15248+++
1`(79,R,8388)@15288+++
1`(80,R,8495)@15409+++
1`(81,R,8753)@15667+++
1`(82,R,8807)@19038+++
1`(82,R,8807)@15732+++
1`(83,R,9042)@15954+++
1`(86,R,9285)@16191+++
1`(88,R,9588)@16477+++
1`(89,R,9617)@16524+++
1`(91,R,9710)@16626+++
1`(92,R,9766)@16671+++
1`(93,R,9811)@20011+++
1`(94,R,9845)@16778+++
1`(96,R,9971)@16886+++
1`(97,R,10026)@16930+++
1`(98,R,10070)@16964+++
1`(99,R,10104)@17002+++
1`(100,R,10142)@17046
```

```
1`(66,B,6268)@13176+++
1`(67,B,6363)@13297+++
1`(68,B,6437)@13331+++
1`(70,B,6584)@13499+++
1`(71,B,6737)@13662+++
1`(73,B,6993)@13898+++
1`(74,B,7083)@14009+++
1`(77,B,7442)@14362+++
1`(79,B,7545)@14462+++
1`(81,B,7746)@14667+++
1`(82,B,7807)@14711+++
1`(83,B,7851)@14786+++
1`(84,B,9094)@19313+++
1`(89,B,8574)@15500+++
1`(90,B,8640)@15538+++
1`(91,B,8678)@15613+++
1`(92,B,8872)@15777+++
1`(93,B,9811)@20011+++
1`(94,B,8975)@15902+++
1`(96,B,9249)@16145+++
1`(98,B,9431)@16343+++
1`(99,B,9483)@16414+++
1`(100,B,9554)@16448
```

```
1`(17,R,1859)@8758+++
1`(18,B,1690)@8614+++
1`(24,R,2512)@9416+++
1`(36,R,3819)@10727+++
1`(37,R,3919)@10816+++
1`(38,B,4057)@14253+++
1`(41,B,3956)@10886+++
1`(43,B,4663)@14900+++
1`(46,R,4869)@11780+++
1`(48,R,5159)@12079+++
1`(51,R,5460)@12375+++
1`(54,B,5045)@11958+++
1`(59,R,6010)@12935+++
1`(60,B,5805)@12724+++
1`(66,B,6268)@13176+++
1`(68,R,6884)@13783+++
1`(71,B,6737)@13662+++
1`(76,B,7219)@14133+++
1`(77,B,7442)@14362+++
```

```
1`(14,R,1398)@14914+++
1`(14,B,1398)@14914
```

Fig. 35.8 Numbered packets received by far-side terminal (Test Case 4)

35.4 Conclusions

Though there no known operational GEO-ground optical systems in use today, it appears that modeling an optical SATCOM system can help determine which architectures can maximize performance in terms of maximizing link availability. To measure this, we determined which alternative architecture minimized lost packets (or duplicate packets for multiple LEO/MEO systems) and minimized the transmission time to send 100,100 IP packets (including retransmissions of lost packets). For our analysis, Case 4 was determined as the best performing, followed by 3, then 2, and then 1. Therefore, recommend optical SATCOM architecture with multiple intermittent LEO satellites. CPNs appear to be well-suited for network and packet analysis since they can be numbered and time-stamped. Latencies and probabilities of transmission success are also useful CPN features. Other conclusions we draw using CPN to model these alternatives are as follows:

- Quality of model matched the quality of measured performance data
- Difficulty in translating what you want the model to do into CPN/Mark-up Language (ML). (Inhibitor arcs often cause deadlocks (unintended consequence)
- One cannot realistically model every factor; focused on the most important ones (latencies, PCFLOS)

As sequels for further research and analysis, recommend the following.

- Implement system that senses PCFLOS along uplink/downlink
- Add two additional alternative architectures and evaluate their performance. These can include the use of multiple dispersed Earth terminals, and/or the use of High-Altitude Long Exploitation (HALE) Unmanned Aerial Systems (e.g., Global Hawk) that can serve as LEO pseudolites, but could remain in PCFLOS regions since they are controllable.

References

1. Version 4.0, CPN Tools ®, downloaded August 2014, www.cpntools.org
2. Petri C (1962) "Kommunikation mit Automaten" (Communications with automation). Darmstadt University of Technology, Darmstadt
3. Reinke D, Forsythe J, .Milberger K, Vonder Haar T (2010) Probability of Cloud-Free Line of Sight (PCFLOS) derived from CloudSat Cloud Profiling Radar (CPR) and coincident CALIPSO Lidar Data. Cooperative Institute for Research in the Atmospherics, Colorado State University, September
4. George Mason University TCOM 607/ECE 699 Course Slides, Lecture 10, April 2013.
5. Calvo R, Becker P, Giggenbach D, Molf F, et al (2014) Transmitter diversity verification on ARTEMIS geostationary satellite. SPIE

Chapter 36
Preference Modeling for Government-Owned Large-Scale Complex Engineered Systems: A Satellite Case Study

Hanumanthrao Kannan, Syed Shihab, Maximilian Zellner, Ehsan Salimi, Ali Abbas, and Christina L. Bloebaum

Abstract The design of large-scale complex engineered systems (LSCES) has been shown to be a distributed decision-making problem involving hundreds or thousands of designers making decisions at different levels of an organizational hierarchy. Traditional systems engineering (SE) approaches use requirements to communicate the preference(s) of stakeholders to drive the decisions of the designers. Requirements, which act as proxies for actual preferences, only state what is not desired of the system rather than what is wanted. This leads to a lack of consistency in the communication of preferences across the subsystems (and even organizations) involved. Also, the current requirements-based SE approaches do not offer any system-level guidance in choosing the best among feasible design alternatives, where all the designs that satisfy requirements are treated equally. Value-driven design (VDD), an alternative SE approach, offers a new perspective on complex system design and emphasizes the importance of capturing true preferences of stakeholders using a meaningful decomposable value function. The formulation of an all-encompassing value function has been proven to be a very tedious process involving a huge overhead, as it requires understanding of the inherent design trades in the system. Past researchers have focused in detail on formulating value functions for commercial endeavors. The primary focus of this paper is to investigate how the formulation of value functions can be approached in a methodical manner using a data-based approach, specifically for a government-based agency (e.g., NASA). More specifically, this paper focuses on formulating a value function for a space telescope mission by identifying and analyzing different aspects involved in capturing preferences.

H. Kannan (✉) • S. Shihab • C.L. Bloebaum
Iowa State University, Ames, IA, USA
e-mail: hkannan@iastate.edu

M. Zellner • E. Salimi • A. Abbas
University of Southern California, Los Angeles, CA, USA
e-mail: mzellner@usc.edu; esalimi@usc.edu

A.M. Madni et al. (eds.), *Disciplinary Convergence in Systems Engineering Research*, https://doi.org/10.1007/978-3-319-62217-0_36

Keywords Preference functions • Value-driven design • Satellite system design • Space telescope

Nomenclature

Bank_{Int}	Monthly bank interest
$B_{\text{spin-off}}$	Benefits from spin-off
Cost_{m}	Monthly cost
n	Predicted number of publications per month
N_{slots}	Total number of observing slots leased over the entire operational lifetime of the telescope
OL	Operational lifetime of space telescope in years
P_{lease}	Leasing price per month
Rd	Discount factor
Rev	Revenue
SI	Science Impact factor
TC	Total Cost
TMC	Total maintenance cost over the entire lifetime of the telescope
TOC	Total operational cost over the entire lifetime of the telescope
V	Value function/value

36.1 Introduction

Large-scale complex engineered systems (LSCES) comprise of several levels of interacting subsystems spanning across multiple levels in an organizational hierarchy, and oftentimes across several organizations**.** The design process involves thousands of individuals making decisions at various levels in the hierarchy, all of which impact the final design. An example of an organization that designs such LSCES is NASA that employed 17308 people as of October 31, 2016. These LSCES are currently being designed using traditional systems engineering (SE) practices, wherein requirements are used to communicate the preferences of the stakeholder [1]. The requirements are delegated down to the designers at lower levels, who then formulate their own requirements and, in turn, pass them down the hierarchy. The subsystems designed by each of the teams are then integrated to form the final system, and if the final design does not align with the stakeholder's preferences, the requirements are reformulated. These requirements, however, only serve as proxies for the preferences of the stakeholder. They do not specify what is actually needed but only what is not desired of the system. This approach does not distinguish between competing design alternatives to choose the best system design. Any design that satisfies the requirements is considered acceptable. Furthermore, the setting of targets might lead to the choice of design alternatives that do not maximize the decision-maker's utility. If an engineer in the hierarchy receives a target that he has to accomplish, he is going to choose the alternative with the highest probability of achieving or exceeding it. Depending on how the target is formulated, his choice might not be the alternative with the highest utility. References [2–4] have assessed and formulated how such targets need to be formulated, so that the best alternative both offers the highest probability of achieving it and maximizes the organization's utility. This paper also mentions common pitfalls that need to be avoided when constructing a utility function.

Multidisciplinary design optimization (MDO) is a field of optimization that emerged from structural optimization in the early 1980s. It differs from traditional requirements-based systems engineering approaches in that it captures and models the couplings inherently present within the system. This ensures consistency in physics when the system optimization is performed. MDO incorporates traditional systems engineering requirements in the form of constraint representations. However, it does not provide a means for creating an objective function, leaving the identification of such a function to the designers.

Recently, value-driven design (VDD), a new systems engineering methodology, has been proposed that uses a decomposable mathematical value function to capture the true preferences of the stakeholder by reducing the requirements [5–9]. These value functions are formed by identifying the fundamental objectives that represent the true preference(s) of the stakeholders involved. Two different definitions of value functions exist, with one arguing that only one fundamental objective exists that can capture the true preference of the stakeholder [10] and the other one arguing that the fundamental objectives can be single or multiple depending on the stakeholders involved [11]. In this paper, the former definition of value functions is used in a first step, assuming that other identified direct value attributes are equal to all design alternatives. With the value function being singular in unit (e.g., Net Present Profit), it can be used to clearly communicate the true preferences of the stakeholder to designers at all levels, thereby enabling consistency in design decision-making and enabling system optimization. Past research has focused in detail on creating value functions in commercial endeavors [5, 7, 12] with only preliminary efforts toward formulating value functions for government-based agencies [13–15]. Past work has identified that the formulation of value functions requires a huge effort and an understanding of the inherent design trades in the system in order to properly capture the true preference(s) of the stakeholder. The focus of this paper is to investigate how the formulation of value or preference functions for government-based agencies can be approached in a methodological manner by using a data-based approach. NASA's space telescope mission is used as a case study in this paper to demonstrate the process involved in creating such preference functions. The next section will focus on a methodological approach to formulating preference or value functions by identifying and analyzing different aspects of the agency involved.

36.2 Methodology

This section focuses on formulating preference functions by identifying relationships that exist between attributes using historical data and finally mapping these attributes to value. As mentioned earlier in the introduction section, this paper uses the definition of value function to be a function of one fundamental objective rather than multiple objectives. Identification of such a fundamental objective requires detailed analysis of the mission, the stakeholders, and their preferences. This analysis is analogous to the process in traditional requirements-based systems engineering approaches, where requirements are defined in a similar manner to express preferences.

36.2.1 Mission Objective

Free from the limitations imposed by atmospheric distortion and light pollution, the two phenomena known to severely impair the optical performance of terrestrial observatories, space telescopes afford the human eye an unobstructed view of the heavenly bodies of near and distant galaxies. From the rich harvest of observations captured by the science instruments on board the space telescope, many long-standing mysteries of our universe were solved and age-old questions answered. However, the number of space telescopes is far from meeting the demand of scientists for observation slots**.** These numbers suggest that scientists can avail an increased availability of observing time offered by new space telescope projects to speed up our rate of discovery, in turn providing more benefit to humankind. Following this logic, we consider the case of a design of a new space telescope as part of a space exploration program undertaken by NASA, a government organization, to reduce the large gap between demand and availability of observation slots.

36.2.2 Stakeholder Analysis

Three major stakeholders of the space telescope project have been identified: NASA, the Congress, and the astronomers' community; each of them has their own set of preferences. Only the primary stakeholders are considered in this paper as they have the most influence and the highest stake on the outcome of the project. Other potential stakeholders might include contractors, the Space Telescope Science Institute, international space agencies, Department of Defense, and other government organizations.

The stakeholder analysis yields their individual preferences but also the respective attributes when they think about making a decision. The decision diagram in

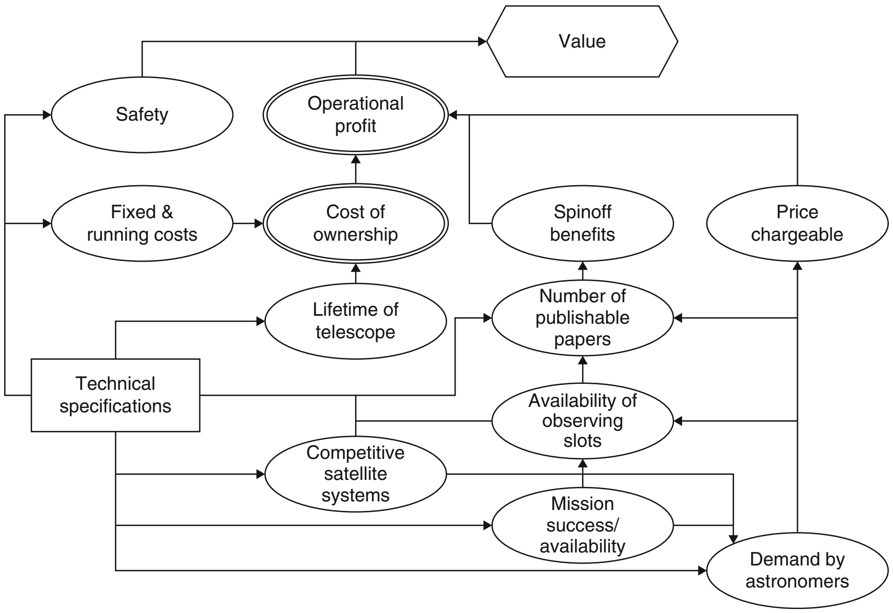


Fig. 36.1 Decision diagram for design of space telescope

Fig. 36.1 offers an overview of the decision situation and captures the preference attributes of all the identified stakeholders.

36.2.3 Preference Analysis

From NASA's mission and vision statement and goal statements for its space exploration programs, it is possible to conclude that NASA would be interested in building a scientific tool which has the highest potential to make breakthroughs in our understanding of the universe and cosmos through astronomical discoveries. In other words, NASA values maximizing the scientific productivity (or return or impact) of the instrument.

Existing space observatories are only able to allocate observing time to a limited number of astronomers. Out of every 1000 research proposals submitted for telescope time, only 200 can be currently accommodated; these numbers suggest that astronomers desire another space-based observing platform providing observing time with capabilities and performance metrics at least as powerful as those of Hubble. Utilizing the talent of this pool of astronomers currently not being served by existing observing facilities is in NASA's interest to advance science.

As Congress is answerable to the taxpayers for its decision to approve funds for the project, it prefers optimizing the profit generated from (or making the most out of) its investment. In order to calculate the revenue, we consider the alternative scenario where Congress decides to direct NASA to lease monthly observing slots of a space telescope from a commercial company instead of granting the funds to NASA to build its own telescope. Assuming that the commercial company wants to make at least as much money per month from the telescope as it would have made from monthly interest if it deposited the capital in a bank instead, it will charge an amount of P_{lease} determined by the sum of the monthly cost and monthly bank interest as shown in Eq. 36.1. So, in essence, P_{lease} is the amount the congress (govt.) is saving each month when NASA is using its own telescope. In other words, when seen from the perspective of the government, this monthly saving P_{lease} (or averted cost) is equivalent to the revenue. The total cost can be decomposed to total investment, total maintenance cost, and the total operational cost over the entire lifetime of the telescope.

$$P_{\text{lease}} = \text{Monthly cost} + \text{Monthly bank interest} \tag{36.1}$$

$$\text{where, Monthly cost} = \frac{\text{Total cost}}{\text{OL} \times 12}$$

$$\text{Total cost} = \text{Total investment} + \text{Total maintenance cost} + \text{Total operational cost}$$

The list of the primary stakeholders and their respective preferences determined from the analyses above are summarized in Table 36.1.

From Fig. 36.1 and Table 36.1, the complexity of the decision situation becomes evident. The decision diagram used in this decision incorporates the decision attributes of each of the project stakeholders, i.e., Congress, NASA, and astronomers/users. For example, Congress's preference of maximizing return from space investment is contained in the deterministic node "Operational Profit," whereas the other preferences can be seen in the uncertainties "Numbers of publishable papers" and "Availability of observing slots." All of the uncertainties and deterministic nodes need to be taken into account by the decision-maker when choosing the technical specifications of the telescope. In order to formulate a sound preference and utility function, common pitfalls have to be considered. Abbas and Cadenbach [16] identifies the most common misuses of utility, which consist of a missing distinction between direct and indirect value attributes, the different preferences of money depending on its origin, using probability as an argument in the utility and preference function, and confining oneself to only multiplicative or additive forms of utility functions. For the decision-maker to be able to assign a distinct preference to each alternative, a distinction between direct and indirect value attributes is necessary. According to [17], direct value attributes are attributes that directly influence the preference of the decision-maker for a certain deal. Indirect value attributes only represent probabilistic relevance to direct values. Thus, preference and value functions are solely constructed for direct value attributes. For the decision situation in the example, the direct value attributes are operational profit and safety, which are used to construct the following preference function:

Table 36.1 List of stakeholders and their preferences

Stakeholder	Preference
Congress	Maximize return from space investment
NASA	Maximize scientific productivity, safety
Astronomers/Users	Maximize telescope observation capabilities and observing time slots

$$\text{Preference} = f(\text{Operational profit}, \text{Safety})$$

This preference function does not include probability as an attribute, which would otherwise be inconsistent with utility theory. For the trade-off that is very likely to exist between the direct value attributes, an additional analysis with the decision-maker is necessary. This deterministic analysis yields a function that might be multiplicative or additive but is not confined to those two types. The resulting so-called iso-preference curves help determine the most favorable alternative under this trade-off situation [17]. The elicitation of the impact of the trade-offs onto each individual preference is not going to be part of the paper, as well as the formulation of multi-attribute utility functions, which are necessary to determine the best decision alternative if the decision-maker is risk-averse or risk-seeking. For further explanations on how to construct utility functions, please see [18, 19, 20]. Choosing the design alternative with the highest preference value ensures that a design is chosen which best meets stakeholder preferences. The specific formulation of the preference function is discussed in the following section and solely focuses on the profit side of the function in a first step.

36.2.4 Preference Function Formulation for Space Telescope: A Case Study

Based on the analysis performed earlier on the mission objective, stakeholders, and their preferences, the formulation of two different preference functions is investigated here. The first one is constructed in such a way that the stakeholder preferences enumerated in Table 36.1 are mapped on to science impact per unit cost, where the science impact factor (SI) is a quantity, less than one in magnitude, derived from the scientific capabilities of the space telescope. In the second preference function, the stakeholder preferences are mapped on to a monetary-based preference function. Here, SI was used to calculate the spin-off benefits.

36.2.4.1 Science Impact Factor Per Unit Cost

In the context of space telescopes, maximizing the scientific discovery is the top priority for NASA and astronomers. But, such a mission would sometimes incur huge costs questioning the feasibility of the mission. The preference function formulated in this section is an attempt to capture the preferences of NASA and the astronomers using a science impact factor and preferences of the Congress using cost. This function, as shown in Eq. 36.2, is a simplistic representation of preferences of the multiple stakeholders involved using a single function. In Eq. 36.2, the SI is obtained by identifying the relationships that exist between the design and technical attributes of the system and finally mapping the attributes to SI.

$$V_1 = -\frac{\mathrm{SI}}{\mathrm{Cost}} \tag{36.2}$$

Some of the widely used metrics to quantify telescope science return are the productivity rate, the number of publications generated, and the discovery efficiency. Of these three metrics, the number of publications in refereed journals, corresponding to the combination of different instrument technologies and mirror characteristics, is first related to the SI as shown in Eq. 36.3. A negative exponential equation is used here to relate the number of papers expected to be published (n) and the SI. Figure 36.2 represents the relationship between the number of papers published and the SI. An exponential equation is used here under a reasonable assumption that the SI increases with the increase in the number of papers published. Specifically, this equation is used to eliminate the SI reaching 1, as it is not realistically possible.

$$\mathrm{SI} = 1 - e^{-an} \tag{36.3}$$

Inside the space telescope system, the mirror collects light from the cosmos and feeds it to the instruments, which then produces a wealth of astronomical observations. The study of these observations by scientists leads to the publication of research papers. So, the choice of instruments, their characteristics, and the mirror characteristics are the key factors which determine the number of publishable papers that can be potentially generated. Based on the existing instrument and mirror data of NASA's past and present great observatories, serving as a priori, a model for the number of papers published (n) is created in terms of instrument and mirror characteristics as listed in Tables 36.2 and 36.3.

The instruments associated with the space telescope (payload) subsystem are categorized broadly in this paper into cameras and spectrographs. Since the attributes associated with each of these categories vary with the instrument type, two different data-based models for the number of papers published are needed**.** These models can be created using any data modeling technique based on the sample data provided in Tables 36.2 and 36.3. A sample of such a model as a function of all the instrument characteristics is shown below.

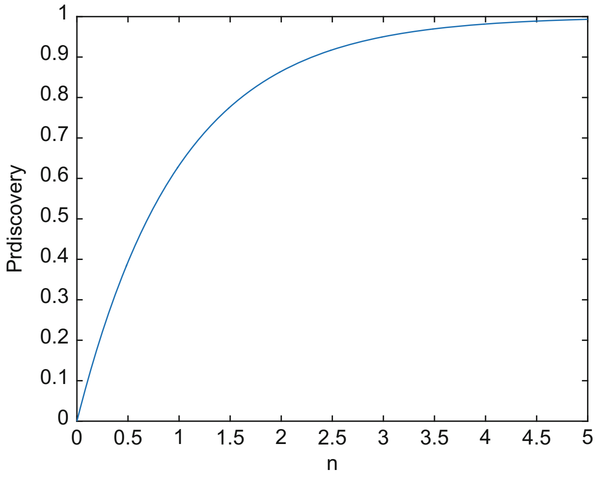


Fig. 36.2 SI vs No. of papers

$$n = \begin{Bmatrix} f(\text{operating wavelength, FOV, Res., Throughput, Mirror size}, \ldots) & \text{For imaging equipment} \\ f(\text{operating wavelength, PS, Res., Active area, Quantum efficiency, Mirror size}, \ldots) & \text{For spectographs} \end{Bmatrix}$$

36.2.4.2 Monetary-Based Preference Function

The previous subsection focused on creating a preference function in terms of SI and cost. The focus of this subsection is on creating a more complicated preference function, where additional terms are added to represent the preferences of each of the stakeholders in a more realistic manner. Since the preferences of each of the stakeholders, as identified in Table 36.1, have different units, expressing everything in financial terms ensures consistency of units and allows rank ordering of system design alternatives. The preference, given in Eq. (36.4), is simply the maximization of value, where value is measured as profit. Here value is calculated as the difference between benefit and cost.

$$\text{Profit} = \text{Benefit} - \text{Cost} \tag{36.4}$$

The two factors contributing to the benefit gained from the space telescope mission are the revenue, which is the product of leasing price (P_{lease}) and number of slots available (N_{slots}), and the science return which is the product of SI and the spin-off benefit ($B_{\text{spin}-\text{off}}$). This can be seen in Eq. 36.5. The leasing price (P_{lease}) is calculated using Eq. 36.1.

Table 36.2 Imaging Space telescope:preference function formulation:instrument characteristicsinstrument characteristics

Science instrument	lamda_l	lamda_u	Pixel size (m^2)	FOV (arcsec^2)	Throughput	Mass (kg)	Power (W)	Cost ($)	No. publications
Chandra ACIS	6.2E-06	1.24E-05	5.76E-10	…	…	…	…	…	11600
Chandra HRC	1.24E-10	1.77E-08	4.134E-11	594	…	…	…	…	25401
Spitzer IRAC	3.6E-06	0.000008	4.5E-10	97344	…	…	…	…	14100
Hubble ACS-WF	3.5E-07	1.05E-06	…	40804	0.43	…	…	…	17900
Hubble WFPC2/Wide field	1.2E-07	1.05E-06	…	16875	0.14	281	80	…	17100
Hubble NICMOS/NIC3	8E-07	2.5E-06	…	2621	391	80	…	…	6210
Compton COMPTEL	4.13E-14	1.65E-12	…	42500000000	…	…	…	…	21100
Compton EGRET	6.2E-08	4.1E-11	…	3450000	…	…	…	…	17500

Table 36.3 Spectrograph characteristics

Science instrument	lamda_l	lamda_u	Pixel size (m^2)	Res.	Active area (m^2)	Quantum efficiency	Mass (kg)	Power (W)	Cost ($)	No. publications
Hubble COS	0.000000165	0.00000035	0.001625	9	0.00065536	10	...	...	...	23900
Spitzer IRS-LW/LR	0.0000139	0.0000399	...	...	...	...	...	...	...	14000
Compton OSSE	1.2398E-13	2.4797E-11	...	...	...	...	...	...	...	28900
Hubble STIS	...	...	...	...	...	...	...	...	...	...

$$\text{Benefit} = \text{Revenue} + \text{Science return} \tag{36.5}$$
$$\text{Benefit} = (P_{\text{lease}} \times N_{\text{slots}}) + \left(\text{SI} \times \text{cost} \times B_{\text{spin-off}}\right)$$

As mentioned in Sect. 2.3, there exists a huge demand for the telescope time. The number of proposals representing the demand is assumed to be 800 in this paper, provided only 200 research proposals are accepted by NASA out of every 1000 proposals. It is also assumed that each research proposal requires a month of the space telescope time. With these assumptions, Eq. 36.6 represents the number of observing slots being used (N_{slots}) over the operational lifetime of the satellite, with each slot representing telescope time for each proposal. This equation states that the maximum slots that can be accommodated are 800 if the demand is not met, and then the number of slots used is given by the product of the operational lifetime (OL) and the number of months per year.

$$N_{\text{slots}} = \left\{ \begin{array}{ll} \text{OL} \times 12, & \text{OL} \times 12 \leq 800 \\ 800, & \text{OL} \times 12 > 800 \end{array} \right\} \tag{36.6}$$

With the revenue component of benefit in Eq. 36.6 being estimated, the monetary value of science return can be calculated by creating a function in terms of SI and spin-off benefits, which is the long-term monetary benefit of the space observatory. It is assumed that with the increase in SI, the monetary benefits due to spin-offs will increase. This is represented using Eq. 36.7.

$$\text{Science return} = \text{SI} \times \text{cost} \times B_{\text{spin-off}} \tag{36.7}$$

It is a common practice to use a value of \$7 to \$14 for $B_{\text{spin - off}}$ (for every dollar invested in space, \$7 is obtained in return). A few examples of spin-offs include charged coupled devices, effects of general relativity in GPS satellites, solar flare safeguard technology in satellites, space/asteroid mining for minerals, averted damage from Earth-bound asteroids, etc. The benefits due to spin-offs can be related to the total cost using Eq. 36.8, which states that for every dollar invested in space, seven dollars are obtained in return.

$$B_{\text{spin-off}} = \$7 \times \text{Cost} \tag{36.8}$$

A block diagram representing the various aspects involved in the formulation of the two preference functions discussed above can be seen in Fig. 36.3. Here DVs represent the design variables that define the design, with the attributes being the system behavior, which are finally mapped to the value through the benefit and cost models.

These value functions can now be used as objective functions to obtain optimal designs. The optimization problem statement associated with the monetary-based value function is provided in Eq. 36.9. Optimizing the satellite system using the proposed value functions will be the focus of our future work.

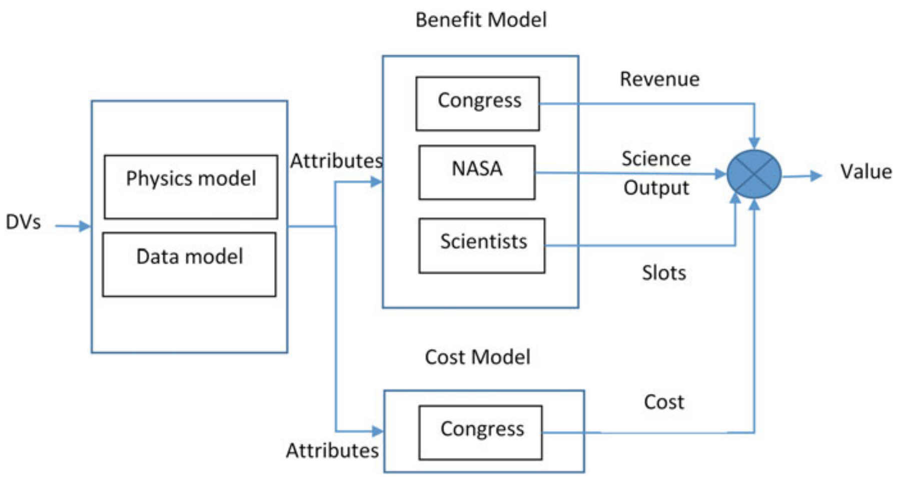


Fig. 36.3 Preference function formulation

Find $\boldsymbol{X} = [\text{Mirror diameter, angular resolution, spectral range, field of view} \ldots ..]^T$

$$\text{Min} f(\boldsymbol{X}, \boldsymbol{y}) = -\text{Profit (or Value)} \tag{36.9}$$

$$\text{Profit} = \text{Benefit} - \text{Cost}$$

36.3 Satellite System

The analysis of the mission objective, stakeholders, and their preferences has resulted in a decision for NASA to build a space telescope that can enable the capture and transmission of observations of the cosmos effectively and in an efficient manner. With that in mind, a satellite model, with a space telescope as the payload, is created in this paper as a test-bed to demonstrate how preference functions can be formulated using a data-based approach. The satellite model is created by modifying the previously developed geo-stationary commercial communication satellite [4]. The hierarchical breakdown of the satellite system is shown in Fig. 36.4.

The satellite system is decomposed into three levels of hierarchy and eight major subsystems at level 1. A total of 31 design variables define the satellite system, out of which 9 are continuous, 1 is integer, and 21 are discrete. A detailed description of the attributes and design variables associated with all the levels of the hierarchy is provided in Appendix A. The attribute-based DSM, shown in Fig. 36.5, represents the organizational (team-based) couplings which are present between the different subsystems at subsystem level 1 (SL1), the first level of the hierarchy. The feedback and feedforward in the attribute-based DSM represent attribute information that

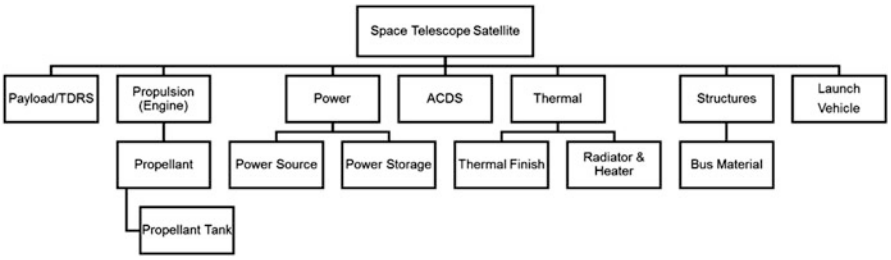


Fig. 36.4 Hierarchical decomposition of satellite

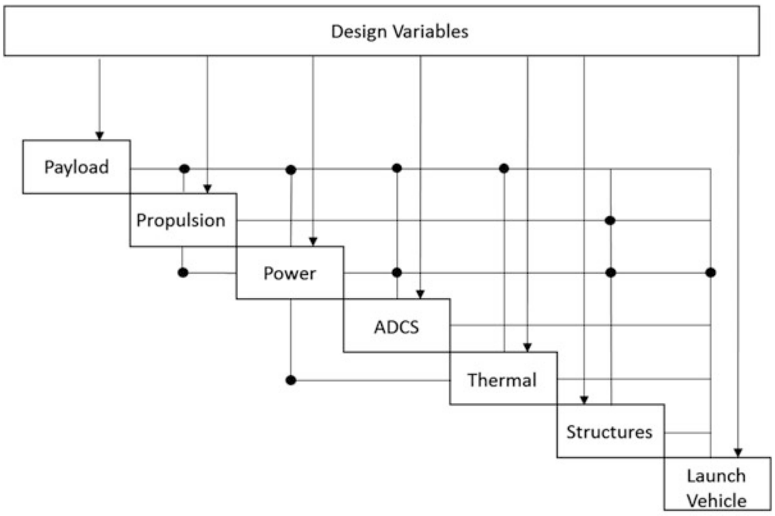


Fig. 36.5 Attribute-based Design Structure Matrix (DSM)

characterize subsystems, like mass and cost of a component or subsystem. Each of these teams in Fig. 36.5 consists of designers that belong to different disciplines. For example, the payload team, which is responsible for designing the payload, includes people from different disciplines like communication, optics, structures, etc., who are responsible for all the analysis associated with each of these disciplines.

36.4 Conclusion and Future Work

With the increase in recognition of design as a decision-making process in the systems engineering community, it is crucial to address the issues associated with inconsistencies due to preferences. There has been a paradigm shift from the traditional requirements-based systems engineering approaches to a value-based approach that emphasizes the need for a more mathematically rigorous representation of stakeholder preferences. It is understandable that one of the major hurdles in adopting a value-based approach is the formulation of a value function as it requires a good understanding of multiple aspects of the system including inherent design trades. With past research being mostly case specific, we perceive that there is still lack of a methodological approach to the formulation of value functions in the design of LSCES. This paper has attempted to introduce such a methodological approach to formulating value functions using data. It has shown some preliminary work in identifying relationships that will enable formulating preference functions. A satellite system with a space telescope as the payload was used as an example to demonstrate how identifying and analyzing the mission objective, stakeholders, and their preferences will enable formulation of value functions. Additionally, this paper focused on the use of a data-based approach to identify relationships that can aid in mapping the system characteristics to value.

Future work will involve obtaining an optimal design for the satellite system using the proposed value functions with the incorporation of more historical data on the instrument characteristics and other aspects of the system. This methodological approach to formulating value functions using data requires analysis with uncertainties. Part of future work will focus on capturing the uncertainties and will also investigate the incorporation of risk preferences using expected utility theory.

36.5 Acknowledgments

This research was funded in parts by the National Science Foundation grants CMMI-1300921, CMMI 16-29752, CMMI 15-65168, and CMMI 16-44991. The opinions, findings, and conclusions stated herein are those of the authors and do not necessarily reflect those of the sponsor.

Appendix A

Table 36.4 Attribute and design variables – satellite system

Levels				Attributes	Design variables
System (Space Telescope Satellite)				Total cost, Revenue	Ground Telescope or Space Telescope
Subsystem level 1	**(SS1)** Payload (Space Telescope)			$C_{payload}$, SNR, Wavelength range, FOV, Resolution, $M_{payload}$, M_{mirror}, $M_{instrument}$, $P_{payload}$	Instrument choice, Frequency, Ant_type, Mirror size, Mirror material
	(SS3) Power			C_{power}	Type of power source
	(SS4) Propulsion			$C_{Engine/kg}$, $C_{propulsion}$	Type of liquid propulsion system(mono/bi)
	(SS5) ADCS			C_{ADCS}	Type of controller
	(SS6) Thermal			$C_{thermal}$	Type of passive thermal control
	(SS7) Structures			$C_{structures}$	Configuration of bus
	(SS8) Launch vehicle			C_{LV}	Launch site, Type of launch vehicle
Subsystem level 2					
	Power	**(SS1)** Solar Array		C_{SA}, Array size, M_{SA}	SA_material
		(SS2) Battery		C_{Batt}, Battery mass, Battery capacity, V_{batt}	Battery type
	Propulsion	**(SS1)** Propellant		$M_{propellant}$, $V_{propellant}$, C_{Engine}, $C_{propellant}$	Propellant
	Thermal	**(SS1)** Surface Finish		$C_{thermalfinish}$	$\left(\frac{\alpha}{\varepsilon}\right)_{SA}$, $\left(\frac{\alpha}{\varepsilon}\right)_{sat,trans}$, $\left(\frac{\alpha}{\varepsilon}\right)_{sat,rec}$, $\left(\frac{\alpha}{\varepsilon}\right)_{bus}$
		(SS2) Radiator and Heater		$P_{thermal}$, $C_{radiator}$, C_{heater}, $M_{radiator}$	$\varepsilon_{rad_{battery}}$, $\varepsilon_{rad_{RW}}$, $\varepsilon_{rad_{proptank}}$
	Structures	**(SS1)** Bus		$C_{bus/kg}$,	Bus material
Subsystem level 3					
	Propulsion	Propellant	**(SS1)** Propellant tank	$M_{proptank}$, $V_{proptank}$,$C_{proptank}$	Propellant tank material

References

1. NASA (2007) NASA systems engineering handbook. vol NASA/SP-2007-6105 Rev1. Washington, DC
2. Abbas AE, Matheson J (2010) Normative decision making with multiattribute performance targets. J Multicrit Decis Anal 16(3–4):67–78. John Wiley & Sons, New Jersey
3. Abbas AE, Matheson JE (2005) Normative target-based decision making. Manag Decis Econ 26(6):373–385. John Wiley & Sons, New Jersey
4. Abbas AE, Bordley R, Matheson J (2009) Effective utility functions from organizational target-based incentives. Manag Decis Econ 30:235–251. John Wiley & Sons, New Jersey
5. Mesmer, BL, Bloebaum CL, Kannan H (2013) Incorporation of value-driven design in multidisciplinary design optimization. In: 10th World Congress of Structural and Multidisciplinary Optimization (WCSMO), Orlando, Florida
6. Andrea H et al (2015) Calculating value gaps induced by independent requirements, deterministic modeling, and fixed targets. In: 56th AIAA/ASCE/AHS/ASC structures, structural dynamics, and materials conference. American Institute of Aeronautics and Astronautics
7. Kannan H, Mesmer B, Bloebaum CL (2015) Increased system consistency through incorporation of coupling in value-based systems engineering. Syst Eng (INCOSE) – Under Review.
8. Collopy PD (2001) Economic-based distributed optimal design. AIAA Paper 4675, p 2001
9. Collopy PD, Hollingsworth PM (2011) Value-driven design. J Aircr 48(3):749–759
10. Hazelrigg GA (2012) Fundamentals of decision making for engineering design and systems engineering
11. Keeney RL (1992) Value-focused thinking: a path to creative decisionmaking. Harvard University Press, Cambridge, MA/London
12. Murugaiyan S, Kannan H, Bloebaum CL (2016) A comprehensive study on modeling requirements into value formulation in a satellite system application – Submitted in CSER, Huntsville, Alabama
13. Bhatia GV, Kannan H, Bloebaum CL (2016) A game theory approach to bargaining over attributes of complex systems in the context of value-driven design: an aircraft system case study, AIAA SciTech, San Diego
14. Goetzke ED, Bloebaum CL, Mesmer BL (2014) Profit and operational-based value functions. In: 15th AIAA/ISSMO multidisciplinary analysis and optimization conference, Atlanta
15. Goetzke ED (2015) Value-driven design of non-commercial systems through bargain modeling. In: Aerospace Engineering, Iowa State University
16. Abbas AE, Cadenbach AH (2016) On the use of utility theory in engineering design, IEEE Systems Journal, New York, Forthcoming
17. Howard RA, Abbas AE (2016) Foundations of decision analysis. Pearson Education Limited, New York
18. Abbas AE, Sun Z (2015) A utility copula approach for preference functions in engineering design. ASME J Mech Des 137(9):1–8. Houston, TX
19. Abbas AE, Bell DE (2011) One-switch independence for multiattribute utility functions. Oper Res, 59 (3):764–771. Catonsville, MD
20. Abbas AE (2009) Multiattribute utility copulas. Oper Res 57(6):1367–1383. Catonsville, MD

Chapter 37
System Safety Data Network: Architecture and Blueprint

Shravan Shett, Mark S. Avnet, and Farzan Sasangohar

Abstract With increasing complexity of safety analysis in sociotechnical systems, there is a need for a mechanism to accurately capture complex information and present it in an easily accessible and understandable form. While there are plenty of accident databases that have been created over the years for specific purposes, a tool that provides a holistic view of all the safety-related aspects of an accident customized specifically per user and industry is largely absent. This paper discusses the conceptual model of the system safety database (SSD), a tool that will offer tailored solutions to multiple classes of users and that will generate reports synthesizing lessons learned from a variety of disparate contexts, providing succinct and actionable information for decision support. The paper also proposes the concept and architecture of a System Safety Data Network (SSDN) that encapsulates a network of safety databases, thereby addressing some of the challenges of a stand-alone safety database. The data network will enable working with structured and unstructured data by integrating multiple relational and NoSQL databases. A full-fledged implementation of the SSDN will enable improved collaboration across industries and corporations. The System Safety Data Network will facilitate analysis across disciplines and contexts, allowing researchers and practitioners to use integrated mixed-methods approaches to conduct investigations, analyses, research, and development activities across multiple levels of a system. The paper also discusses the steps involved in the implementation of such a data network and the challenges involved. In addition, the current work in data categorization and interpretability of incident data is discussed. When completed, the System Safety Data Network will provide stakeholders at all levels, from individual operators to policymakers, with the tools and perspectives needed to improve the safety of complex sociotechnical systems.

Keywords System safety • System safety data network • System architecture • Accident investigation • Accident case studies

S. Shett (✉) • M.S. Avnet • F. Sasangohar
Texas A&M University, College Station, TX, USA
e-mail: shravanshetty@tamu.edu; avnet@tamu.edu; sasangohar@tamu.edu

A.M. Madni et al. (eds.), *Disciplinary Convergence in Systems Engineering Research*, https://doi.org/10.1007/978-3-319-62217-0_37

37.1 Background and Motivation

With advancements in models and analytical methods used in accident investigations [1], it is now generally accepted that accidents in sociotechnical systems are rarely due to a single isolated cause [2]. Let use consider the Swissair Flight 111 accident as an example. Though the case for the accident was reported as a fire caused by faulty wiring, a convolution of lack of clear regulations regarding in-flight fire control, poor crew training, and highly flammable thermal insulation blankets led to fire expanding and engulfing the cockpit [3]. This emphasizes the importance of interrelationships between sub-events and subsystems and the necessity to capture and analyze such complex information to develop a holistic understanding of accidents. The value and learnings from such information would arguably increase if it was easily available across domains and industries in a single central repository.

Safety databases have been built and managed by many organizations, such as the Aviation Safety Network (ASN) and the National Aeronautics and Space Administration (NASA), which contain reports on thousands of accidents [4, 5]. But the values drawn from these reports are usually limited at best, as a systematic and in-depth analysis of accidents is scarcely conducted and the data available is rarely easily accessible. There is no dedicated tool to analyze and integrate the learnings from available data and apply its knowledge to benefit safety across all industries.

In our previous work, we describe the system safety database as a universal repository of information about accidents, regulations and regulatory bodies, expert analyses, and safety methods and frameworks across various industries [6, 7]. Systematic analysis of cases using the multilevel frameworks [8] and case-based reasoning approach [9] was conducted on seven individual cases. During the course of research, the team identified that the challenge to a universal repository of accidents was a two-phased problem: unraveling the depth and complexity of information by obtaining data and creating a collaborative environment for sharing safety information. One of the most important limitations of the suggested model was that safety information or accident data is not easily available, especially with incidents related to government organizations or corporate entities, and a system providing incentives and encouraging collaboration needed to be developed. Though much of the data in the civil sector (in the United States) has been made available through the Freedom of Information Act (FOIA), the bureaucracy involved in submitting and responding to a FOIA request presents a challenge. In the defense sector, much of the data is classified and often cannot be shared externally to the Department of Defense (DoD). In the private sector, companies often are incentivized to respond to regulations by ignoring or hiding information about safety incidents rather than using that information to reduce the chances of future accidents. For this reason, these companies are often unwilling to divulge the data needed for a repository such as the SSDN. Hence, convincing prospective partners about the security and advantages of a data network and emphasizing the issues of control on data collaboration information is critical to the success of the project.

The goal of this paper is to describe the architecture and blueprint of a System Safety Data Network (SSDN) and to provide an update on the implementation efforts. The SSDN is designed to a be a central, service-based data network that, in addition to the features of the SSD, provides the ability of secure and controlled data collaboration for third-party collaborating organizations. The internal databases will host systematically analyzed information from a broad spectrum of industries and will be able to generate customized reports for particular stakeholder groups and system contexts. In addition, the Application Program Interface (API) services provide collaborating organizations with the ability to incorporate analytical and data services into their personal systems while providing methods to contribute controlled, anonymized, and secure content to the SSDN at their consent. The paper also describes the challenges and opportunities provided by a full-fledged safety data network.

37.2 Methodology

To effectively capture the data to build a repository of accident information, a set of 16 comprehensive semi-structured interviews with industry experts on safety across a variety of application domains and disciplines was conducted. Participants were sampled using maximum variation (Patton 1990) based on their areas of expertise to ensure a broad base of information. The interviews were audio-recorded and transcribed. All interviews were based on a common set of questions in a preconstructed interview guide. The interview guide contained questions on the participants' background and their relative knowledge and experience with safety incidents, and also open-ended questions on any summary thoughts or perspectives that they would like to add. Further context-based probes were used to gain additional information, clarification, or expansion as needed. To transcribe the audio files, transcribers listened to each audio transcript and typed the data into a file. A back-referencing technique was used until each file was fully accurate. Upon completion of the interviews, results were analyzed and a list of relevant attributes was developed. The data collection was then split into two steps involving case analyses. Initially, to gather a wide range of data for the repository, maximum variation of accident industries was used to get extensive information about 40 cases. The cases were carefully chosen as to cover a variety of domains and industries. Next, seven cases were selected from this set for in-depth analysis based on the content available in public domain. A method for in-depth analysis to pick relevant information from cases was developed using the following three requirements:

- Each case analysis must contain enough information that a randomly selected, uninformed user could grasp what caused an accident with a reasonable level of understanding.
- While analysis must be detailed, each analysis must also be precise and compact in size. Given the full scope of the database and the volume of data included in the final version, the contents of each analysis must be as short as possible to conserve database memory. In practice, this means accident summaries are generally limited to a paragraph.
- Each analysis should strive to use primary sources when possible. Accident reports from regulatory organizations are preferable. As opposed to primary sources, secondary sources often limit the scope of an accident and do not include the depth required to pick out underlying causes. Also, secondary sources are not always archived, meaning that a source cited in the database might be impossible to find after a given period of time, leaving entries in the database unverified.

Next, an in-depth analysis of the collected cases was performed by in-house researchers. The underlying causes for an accident were identified, and each underlying cause was listed and classified as being on a technical, human, organizational, or societal level. Dissecting cases to uncover underlying causes is a time-consuming process, so efforts were directed at fully analyzing cases that expand upon the information derived from other cases. From the data gathered by systematic analysis of seven cases, an entity relationship diagram of the internal database was created. The diagram was then modified to build on the important information specific to individual cases. The class diagrams were then developed, and individual attributes were mapped to the data architecture. An alpha version of the database was built and data was loaded into the database. Test queries were conducted on the database to ensure that the data retrieved was of expected format and quality. Drawbacks of previous attempts at safety databases were studied. Based on analysis of feedback and lessons from previous databases, the idea of distributed architecture was proposed. After analyzing technical feasibilities to store required data structures, advantages, and disadvantages, an alpha version of the System Safety Data Base was developed.

37.3 System Architecture

One of the primary concerns of a collaborative central repository of safety information based on previous attempts [4, 5] is the reluctance of corporations or government organizations to share accident information due to the secure nature of the information, or in some cases the tendency to hide past mistakes. Hence, the Architecture of the SSDN was developed focusing on data security and feasibility of collaboration as primary factors. The design consists of a central core that is in the public domain containing information analyzed by researchers and those made available by willing private organizations. To promote sustenance and accuracy of information in the system, the feature for data insertion from individual users with valid proofs is envisioned. The system is designed to motivate whistle blowers and anonymous users, making the most accurate information available in the public domain. Organizations could then develop their exclusive nodes and interfaces. The participating organization will have access to all the information available in the public core along with their own proprietary node and, when possible, and sanitized information from other organizations' proprietary nodes whenever feasible. The organization would be in control of data pertaining to its proprietary node and can work with in-house researchers to sanitize and anonymize information to be shared to the public core.

The SSDN's front end can benefit from a web service providing users a rich and responsive experience. The data will be provided dynamically by a backend Communication Management Interface (CMI), which is the central data management component of the system. The unit also manages secure logins by looking up into an encrypted login database containing user information. The CMI is also connected to a series of internal and external APIs through which content request and responses are conducted. The APIs are connected to multiple database management systems (DBMSs) through which the requested information is processed. Here the requests are broken down into queries, and the data are retrieved from the databases. The databases themselves are distributed across relational and NoSQL databases. The relational part captures the structured, surface-level information and also defines the relations between data components. The NoSQL part captures all unstructured and complex information in its entirety. This architecture enables the storage of both structured and unstructured forms of data.

To illustrate how the System Safety Data Network can be used, we use a recent accident: the disappearance of Malaysia Airlines Flight MH370 [10]. On March 8, 2014, flight 370 from Kuala Lumpur to the Chinese capital, Beijing, lost contact with air-traffic controllers and since then several investigations into the accident have taken place. Using the SSDN (Fig. 37.1), a user investigating the accident would make a request on information available through the SSD front end (Section 1). The CMI/LV receives this request, verifies the user, and looks up the data dictionary for the location of the relevant data (Section 2). The data might in this instance be a culmination of information from our internal database, the Department of Civil Aviation Malaysia's database, Federal Aviation Administration (FAA) database, and NASA's Aviation Safety Reporting System Database. Based on the information in the data dictionary, the CMI/LV makes API requests to relevant systems (Section 3 and Section 5). The requests are processed by the systems independently as per its business logic, and relevant information is returned in a predetermined response format. This data is loaded back to the front end dynamically completing one request cycle.

The service-based nature of the architecture has the following strengths:

- Data available to the user will be in the most up-to-date form of information as the information is obtained at runtime via API requests from multiple relevant

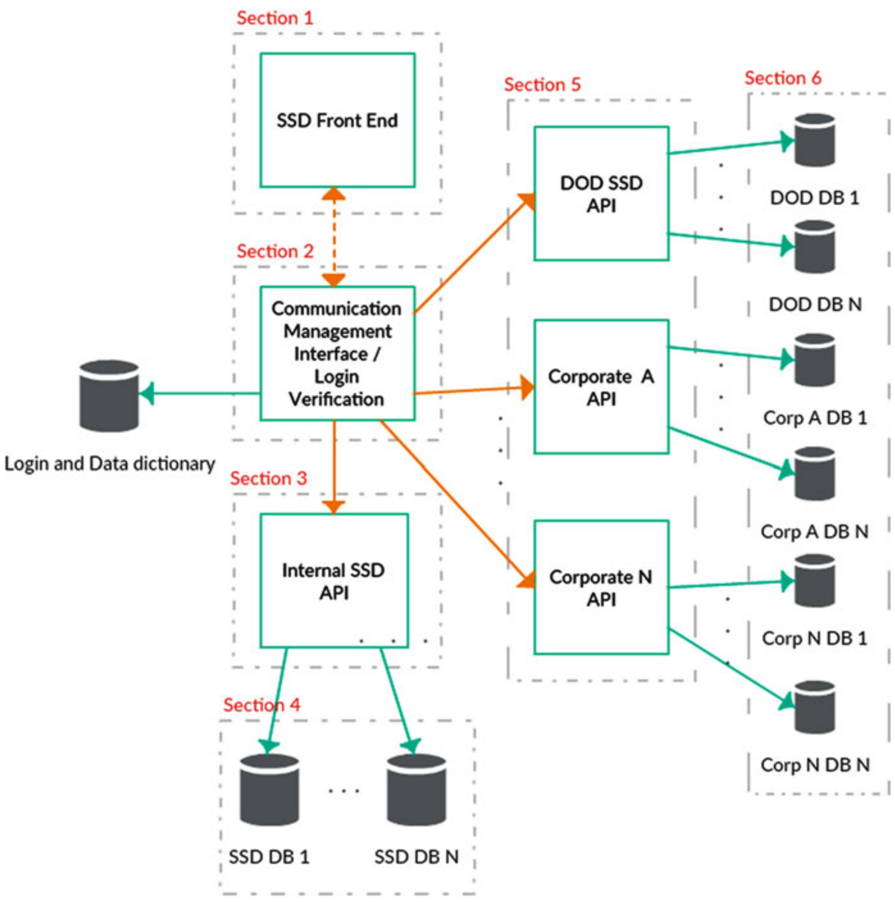


Fig. 37.1 High-level visualization of SSDN's modular architecture

sources as opposed to having prescheduled batch jobs pulling in information into internal databases. For example, if information on the Malaysian Airlines case is updated in the NASA database, traditionally, this data will not be reflected onto the central repository until it is updated in the internal database by a scheduled job. But with a service-based design, the data updated in the NASA database is immediately available for users in SSDN.

- Data security, ownership, and availability will be controlled by third-party partners, improving trust in the collaborative effort while decreasing liabilities on sensitive data. A service-based architecture provides control of data to the collaborating partners, decreasing security concerns while providing a stepping stone for future sharing of anonymized data to the internal databases. Such an architecture would specifically improve the possibility of collaboration by corporate or government organizations. The modular nature of the system enables the system to have high availability. Since a request will involve calls to multiple sources, the probability of all the sources becoming unavailable simultaneously would be very low.

Such distributed architecture, however, has several important limitations:

- There is limited control on responsiveness of the system since the response of the systems depends on responses of collaborating systems and may suffer as request load increases. Also, with the growth of the data network and the addition of new systems, responsiveness may become sluggish.
- Though there is no single point of failure, full availability of the system will be affected if one of the collaborating systems becomes unavailable. Data specific to that partner becomes non-accessible unless redundancies are built into the collaborating system by the partners.
- Also, because partners are responsible for system maintenance, partners' willingness to engage dedicated resources for the maintenance of such collaborative system is uncertain.

In the next section, we discuss the blueprint for a successful SSDN and propose a method to visualize dependencies among its various phases.

37.4 Blueprint for a System Safety Data Network

The System Safety Data Network concept is still in its infancy and considering the scope of the project, there are a lot of white space risks to be accounted for. In this section, we provide a high-level methodology for the implementation of a full-fledged System Safety Data Network (Fig. 37.2). We discuss the scope of the work in progress and various intermediate objectives of the work.

Step 1:

Analyze and decompose the accident cases available in the public domain and build an internal collection of accident information. This process is implemented by in-house researchers. The data extracted by analyzing accident cases is discussed in [7]. The basic data catalog has the following structure:

- The date of the accident
- The number of fatalities or injuries
- Monetary damages (if information is available)
- The proximate cause of the accident
- The duration of the accident
- The responsible organization(s)
- The industry of the organization involved in the accident
- Any regulatory organizations entrusted with safeguarding the responsible organization(s)
- Underlying causes for an accident
- Classification of the underlying causes

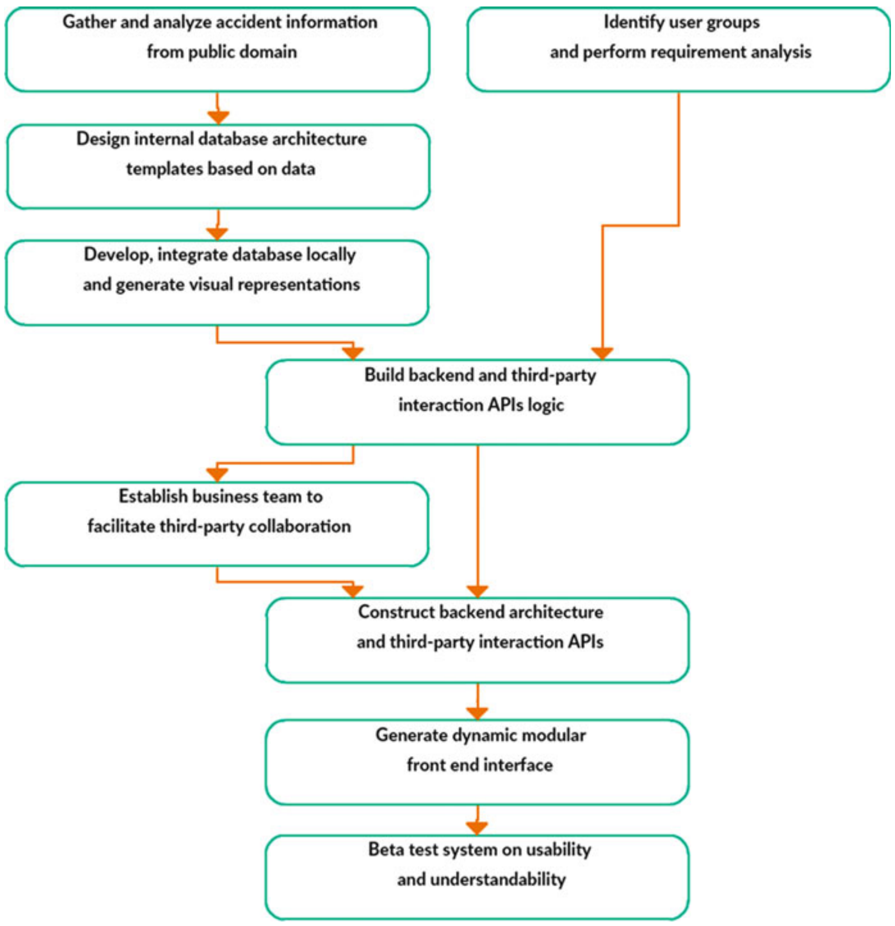

Gather and analyze accident information from public domain
Identify user groups and perform requirement analysis
Design internal database architecture templates based on data
Develop, integrate database locally and generate visual representations
Build backend and third-party interaction APIs logic
Establish business team to facilitate third-party collaboration
Construct backend architecture and third-party interaction APIs
Generate dynamic modular front end interface
Beta test system on usability and understandability

Fig. 37.2 Blueprint of phases involved in the fulfillment of the SSDN

Step 2:
Build internal database architecture templates to accurately capture the diverse structured and unstructured data captured during the analysis and extraction phase. This phase involves evaluating the structure of the available data, analyzing the various complex interrelations between these data structures and building an architecture that accommodates and integrates these relationships.

Step 3:
Build an alpha version of the internal databases on a local server and implement the integration of the diverse databases. This phase could benefit from network representation of complex accident information. In a recent effort, the sources of failures in complex accidents and their relationships were captured in a visual format to help users understand the complex nature of several nuclear accidents and to easily map the interrelations and major factors [11].

Step 4:
Identify user groups and build a core group of prospective users to understand the needs of each user group and map requirements in collaboration with these prospective users. This phase involves market research by conducting interviews, surveys, and polls to build a comprehensive set of requirements for each group of users interacting with the tool. Based on the analysis of these requirements, use cases and activity charts are developed to map the functionality of the system.

Step 5:
Build the backend architecture based on the research done in step 4, and wrap the internal functionality in modules and expose the required interface methods using APIs. This step involves building the business logic of the system using a modular approach and integrating the internal API modules to backend databases. Also, this step involves documenting and building a demo third-party interaction API, which will act as a reference and provides expected communication protocols to the data collaboration partners. We acknowledge that the implementation of the partner APIs will have to be customized on a case-by-case basis depending on the architecture and structure of the collaborating partner systems.

Step 6:
Create a business team in charge of growth of the data network by approaching corporations and government organizations for data collaboration using APIs. This will be a crucial phase in the development of the database, as convincing reluctant parties to share data is one of the biggest challenges that would likely hindered the development of a centralized data repository.

Step 7:
Build a responsive, maintainable, and dynamic front-end interface for the SSDN that interacts with the back end via APIs. A modular design will help with incremental addition of features to the system, thereby providing flexibility for upgrades.

Step 8:
Beta-test the system with a core group of trial users, collecting data on usability and interactivity of the systems features. A feedback loop and an active user community will sustain the continuous development of the tool.

The abovementioned blueprint for the implementation of a SSDN would set the stage for extensive data analysis on accidents and contribute to a deeper understanding of accident complexities across industries.

37.5 Current Work-in-Progress

The database architecture capturing the surface-level information has been completed, and currently work is being conducted in mapping the complex interrelation between causes and capturing the holistic view of the accidents. The teams started exploring NoSQL databases and are currently looking into suitable options for the application. An alpha version of the relational database was built on MySQL and data from the analyzed accidents were loaded. A preliminary market research on the user groups was conducted, and based on the requirements gathered from the research, basic use cases for the system were developed. After consulting industry experts and analyzing the challenges in the development of the system safety database, the idea of a data network was proposed. The current architecture was designed after analyzing the strengths and weaknesses of all the proposed architectures and conducting a suitability analysis mapping requirements against the strengths and weaknesses of models.

37.6 Conclusion: Challenges and Opportunities for the System Safety Data Network

In this paper, the need for a comprehensive collaborative central repository containing information on the underlying causes of accidents, regulations, safety analysis methods, and experts was discussed. The system would be open source and would also accept verified contributions from anyone that wishes to add information to the system. It would also contain collaboration partners that would contribute to the repository by opening screened and anonymized sections of their internal data to the network via API services. The paper further discusses the architecture of such a data network and presents the blueprint to building the database. It also provides an update on the current work, status, and challenges of the project. The applications of a fully functional data network include understanding of risk factors in the system, enumeration of relevant regulations, collaborative work space for research, and many more. Once completed, the System Safety Data Network will provide analytical tools and generate tailored reports for all stakeholders to measure and make data-driven decisions, thereby improving safety of complex engineered systems.

Considering the scope and complexity of the project, white space risks need to be accounted for and the blueprint needs to be flexible to incorporate changes due to requirements or challenges that occur during implementation. A thorough analysis of strengths, weaknesses, opportunities, and threats (SWOT) needs to be conducted, and a business plan for the system needs to be developed. The current approach is to take the initial steps in analyzing databases and collecting data from specific industries and building on top of the in-house knowledge of the incidents. The bulk of the future work involves addressing the challenge of mapping the complex interrelations to data structures on a case-by-case basis. This includes unifying and mapping accident, regulations, system analysis methods, and export information and creating a method for generating basic safety checklists using available information. Comprehensive and robust definitions of complexity, scale, and scope of accidents need to be standardized, and visualization mechanisms to incorporate more complex information in an easy to understand manner must be developed.

Sustainment and maintenance of the network over a period of time is achieved by maintaining the public core as open source and commercialization of the private nodes. Open source of the central repository promotes an active user base committed to maintaining safety information accessible and accurate. To achieve sustainment, use cases on user data insertion and automated validations through dynamic channels need to be incorporated into the design of the system. Allowing users the ability to insert or update case information provides an alternate channel of undocumented information on the incident into the system. The design of the system should empower and motivate users to be the flag bearers of an open, accurate, and collaborative safety information data network.

With respect to maintenance of the system, upon completion of analyzing sufficient accident data available in the public domain and development of the core public node, the collaborating organizations build and maintain their own proprietary nodes with reference from the internal team managing the demo third-party interaction API. For the maintenance and update of the core central components, a nonprofit organization or board would be required and would have exclusive responsibility to keep the information accurate and accessible. A systematic survey of available organizations with the right motivation, vision, and commitment to accessible safety information needs to be conducted. Tackling these challenges regarding the SSDN will go a long ways in the creation of a central repository of safety information.

Acknowledgments The authors would like to thank the 16 expert researchers and practitioners that leant their time and expertise to participate in interviews for a related study and whose ideas contributed to the foundation for the database. Early development of this research was supported by the Systems Engineering Research Centre (SERC) under an RT 128 research incubator grant from the Office of the Deputy Assistant Secretary of Defense for Systems Engineering (ODASD (SE)) project. The authors would also like to acknowledge Nanditha Soundararaj (MIS, Texas A&M University) for her input and support in finalizing the paper.

References

1. Oakley J (2012) Accident investigation techniques, 2nd edn. American Society of Safety Engineers
2. Public Education and Conference Section. In: Fixing the system with root cause analysis, Oregon Occupational Safety and Health Division
3. McDonnell D, Swissair Transportation Limited (1998) In-Flight fire leading to collision with water. McDonnell Douglas MD-11 HB-IWF Peggy's Cove, Nova Scotia 5 nm SW
4. Aviation Safety Reporting System Database. NASA. Accessed 8 Nov 2015
5. ASN Aviation Safety Database. Aviation safety network. Accessed 8 Nov 2015
6. Faucett C, Shetty S, Avnet SM (2016) System safety database: challenges and opportunities. In: CESUN Conference
7. Shetty S, Faucett C, Avnet SM (2016) System safety database: use cases and applications. In: ISERC Conference

8. Avnet SM, Smith-Jackson LT (2015) A multilevel framework of system safety: technical failures, human factors, organizational behavior, and societal influence. T.A.M. University, Editor: College Station, Texas (in White Paper)
9. Bergmann R, Althoff KD, Minor M, Reichle M, Bach K (2009) Case-based reasoning – introduction and recent developments. German Research Foundation
10. Malaysia Airlines Struggles to Salvage Its Image a Year After Flight 370 Disappearance. In Time Magazine, 201
11. Sasangohar F A holistic investigation of complexity sources in nuclear power plant control rooms. Masters Thesis, MIT

Chapter 38
Scalability in Self-Organizing Systems: An Experimental Case Study on Foraging Systems

James Humann, Yan Jin, and Azad M. Madni

Abstract Scalability is a great advantage for systems that face uncertain demand. Scalable systems can be increased in size at a reasonable cost to meet increasing demand, or they can be reduced in size to minimize ongoing costs in the face of falling demand. Self-organization is often hailed as a strategy for creating scalable systems, as they have low integration costs and no communication bandwidth limit from a central controller. This paper investigates the scalability of a self-organizing foraging system. The results show that there are fitness penalties associated with scaling systems up or down from the size they had been optimized for, and these penalties are higher for scaling up rather than down. However, if the system's agent behavioral parameters can be adjusted as the system size changes, the system-level fitness increases linearly with size.

Keywords Scalability • Self-organization • Agent-based modeling • Multi-agent simulation

38.1 Introduction

38.1.1 Scalability

Dynamic allocation of resources is an important strategy in uncertain environments. Scalable systems can change in size to meet changing requirements [1], grow when they face greater demand, and shrink to meet falling demand with fewer resources. A rigid system that is not scalable may incur costs if it is undersized or oversized. These costs include the excessive costs of building a system's capabilities beyond their demanded performance, ongoing maintenance of a larger system, or opportunity cost of not capitalizing on higher than expected demand. This places enormous

J. Humann, PhD (✉)
Intelligent Systems Technology, Inc., Los Angeles, CA, USA
e-mail: humann@usc.edu

Y. Jin, PhD • A.M. Madni, PhD
University of Southern California, Los Angeles, CA, USA
e-mail: yjin@usc.edu; azad.madni@usc.edu

A.M. Madni et al. (eds.), *Disciplinary Convergence in Systems Engineering Research*, https://doi.org/10.1007/978-3-319-62217-0_38

pressure on the system's engineer to design the system's capacity so that it is just right, but this may not be a realistic expectation for complex environments or long-life-span systems.

38.1.2 Self-Organizing Systems

Self-organization is one strategy for designing systems that can scale to their required capacity at run time. A self-organizing system is made of a group of interacting autonomous agents that are not subject to any central controller. Integration costs in self-organizing systems are low, so adding capacity is only limited by the cost of the hardware. Their distributed nature also eliminates a possible upper limit on system size that could be a constraint of a central controller. An inspiration from the natural realm, locust swarms, can grow in size up to 10^9 insects [2] yet still fly as a cohesive flock without any single locust leading or coordinating the swarm. Two complementary forces have recently increased the importance and visibility of self-organized architectures: a market pull (from customers requiring adaptability, scalability, and resilience in systems) and a technology push (from enabling technologies such as miniaturized robots, bio-inspired robotics, and the Internet of Things) [3].

Self-organizing systems rely on local communication and sensing, so their control schemes have inherent potential for scalability. A centralized controller would have to read and synthesize all of the data available from lower-level sensors [4]. As the size of the system increases, so do the demands on the central controller until it is pushed beyond its capacity, which would cause it to crash or at best delay its outputs or ignore some information. In self-organizing systems, agents form dynamic local networks that are relatively stable in size and may be loosely linked to one another. As long as the agents are properly designed to analyze the data available to them locally, more can be added to the system without stressing a system-wide bandwidth limitation. Despite the local frame of action, the interactions among agents can spread information system-wide and enable complex system-level behavior [5]. Thus scalability is often hailed as a promising feature of self-organizing systems [5, 6], but naively scaling systems without understanding the possible pitfalls may lead to system failures [7, 8].

38.1.3 Related Work

Ross et al. give a formal, domain-independent definition of scalability as a specific type of changeability, among other "ilities" such as adaptability and modifiability [1]. In [5, 9] it was shown that large scalable vehicle networks could be formed from peer-to-peer communication. This network could disseminate safety and traffic information orders of magnitude further than the peer-to-peer transmission range without relying on or being limited by any third-party infrastructure.

Self-organization has been suggested as a strategy for scalable formation control for pedestrians and vehicles [10, 11]. In fact, the Boids algorithm, a famous self-organized flocking algorithm, became popular initially because it was so computationally efficient in its ability to display computer-generated flocks of birds [12]. Self-organized foraging in robotic systems has been demonstrated in [7, 13] where groups of robots gathered pucks and boxes into a central location without directly communicating.

This paper is also the continuation of a series on cellular self-organizing (CSO) systems, so called because each agent in the system is rather simple, but by working together, the agents can display complex behavior, like the cells in a human body. CSO research has demonstrated reconfigurability through information sharing [14], field-based control for searching and swarm formation [15, 16], and applications in exploration, box pushing, and protective tasks [17, 18]. The two complementary goals of CSO research are to understand self-organization in natural systems and to apply this knowledge to the design of engineered resilient systems [19].

38.1.4 Agent-Based Modeling and Genetic Algorithms

In [20, 21], we introduced a methodology for the design of self-organizing systems. The methodology relies heavily on the use of agent-based modeling for system analysis and genetic algorithms for optimization. Agent-based modeling treats elements of the system as autonomous actors with sensing, reasoning, and decision-making capabilities. The interactions of the agents can be simulated and tested on a computer to study emergent properties of the system. This approach allows the engineer to focus on a small-scale problem: accurately defining agent behavior. The larger-scale problem, determining the complex results of agent interactions, is left to the computer. Genetic algorithms (GA) have been used in previous work [22, 23] on the design of multi-agent systems because they are efficient algorithms for optimizing large, complex, and noisy search spaces. Briefly, a genetic algorithm operates on a population of potential solutions (agent behavior settings) by scoring them with a fitness functions and improving them in successive generations [24, 25].

Combining the two approaches allows a designer to focus on the conceptual design of the agent behavior. As long as it is parameterized, creating a class of systems, the genetic algorithm can search for an optimal point solution by repeatedly invoking the multi-agent simulation software.

38.1.5 The Foraging Task

In a foraging system, agents must find a resource and transport it back to a base location. One of the most famous examples is found in nature: the food foraging behavior of ants [26, 27]. Ants begin searching for food around their nest individually. When one by chance finds food, it lays down a specific pheromone (chemical scent) as it carries the food back to the nest. Other ants then randomly find the pheromone trail, follow it to the food source, and lay down even more pheromones in the same location, causing a positive feedback loop of increasing pheromone concentration. Eventually, the whole colony is recruited to exploit the food source, forming an efficient straight line between the food and the nest. This ant behavior is actually so adept at solving search problems that have been abstracted into an optimization algorithm, known as ant colony optimization [28].

In this paper, the foraging behavior of ants serves as an inspiration for the design of an artificial self-organized foraging system. In practical applications, the basic pattern could be seen in a system that finds and gathers waste in a cleanup task, a search and rescue system, or a system that harvests crops.

38.2 Experimental Setup

38.2.1 Foraging Task and Simulation

Figure 38.1 shows the initial setup of the simulated foraging task in NetLogo [29]. The food is marked in green, and the home base is marked in red. The objective is to maximize the amount of food returned to home within a time limit. Agents can sense food and other agents within 3 pw (pw is the width of a "patch" in the simulation world; the size of a patch can be seen in the blocks forming the home base). When an agent moves onto a patch that contains food, it extracts five units of food and changes its own color to green, signifying to other agents that it has found food. If it carries food back to the home base, it deposits the food and changes its color back to brown. This stored food then counts toward the system's fitness score. Agents maintain no memory of the food location and must find it anew every time they leave the home base. They can only sense it when it is within their 3 pw detection radius. Agents can, however, sense the direction toward home at all times.

38.2.2 Agent Behavioral Model

The agents in this system have two states: carrying food or not carrying food. They have 18 state-based behavioral parameters, summarized in Table 38.1.

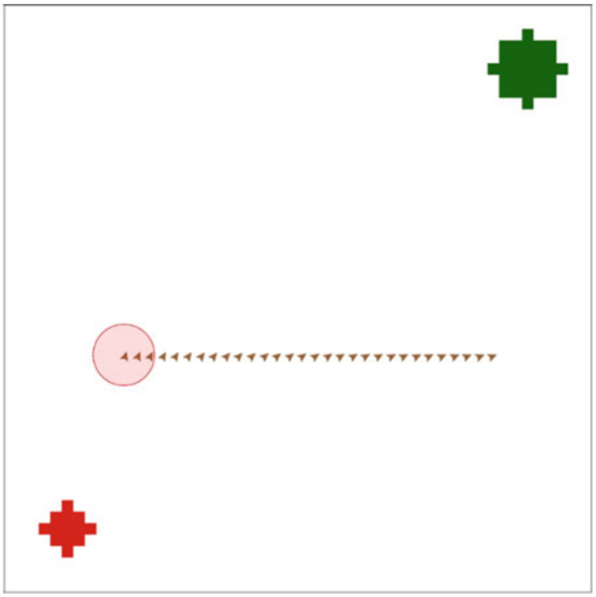


Fig. 38.1 Initial configuration of a one-row foraging simulation. The *red circle* indicates the detection range of an agent

Table 38.1 Foraging behavioral parameters

Agent state	Neighbor state	Cohesion	Avoidance	Alignment	Randomness	Home	Food
Food	Food	C_1	O_1	A_1	R_1	H_1	F_1
	No food	C_2	O_2	A_2			
No food	Food	C_3	O_3	A_3	R_2	H_2	F_2
	No food	C_4	O_4	A_4			

Notionally, cohesion is an agent's desire to move toward its neighbors; avoidance is its desire to move away from its neighbors; alignment is its desire to match speed and direction with its neighbors; the randomness desire changes at each time step; home is the desire to move toward the home base; and food is the desire to move toward sensed food. These desires are all considered simultaneously, and a weighted average of the stimuli is used by the agent to decide on its next step.

To aid in this decision process, field-based behavior regulation [19, 22] is used. Field-based regulation treats all stimuli as sources or sinks in a mathematical field. In this paper, agents consider two fields: a task field of stimuli in the environment and task and a social field of other agents. This separation is used to aid the designer, as the social field can create system structure while the task field deploys it in space. Agents calculate the field value at every reachable point in their immediate vicinity and step to the point with the highest field value. The field equations are given as

$$FLD_S(r,\ \theta,\ \phi) = C \cdot \frac{-1}{N}\sum_{i\in\eta}^{N} r_i + O \cdot \frac{-1}{N}\sum_{i\in\eta}^{N}\frac{1}{r_i} + A \cdot \frac{1}{N}\sum_{i\in\eta}^{N}|v_i|\ \cos(\theta-\phi) \tag{38.1}$$

$$\mathrm{FLD_t}(f,\ h,\ \phi) = S_{\max} \cdot F \cdot (\cos(f-\phi) + H \cdot \cos(h-\phi)) \tag{38.2}$$

where η is the set of N agents in the calculating agent's radius of detection; r_i is the distance from a point to the agent's neighbor; ϕ is the point's angle away from the agent's current heading; θ is the neighbor's current heading relative to the agent; $s_{\max}$ is the agent's maximum step size; f is the angle toward food; h is the angle toward home; and C, O, A, F, and H are state-determined parameters as described in Table 38.1. The results of Eqs. (38.1) and (38.2) are added together to calculate the field value of any point.

38.2.3 Simulation and Optimization

The 18 behavioral parameters given in Table 38.1 define a class of systems. Any set of particular parameters fixes the behavior of that system. This allows the GA to search through a space of possible systems for optimal behavior. The GA's fitness function is given as

$$\text{fitness} = \text{food}_{\text{r}} + \frac{1}{N}\sum_{i=1}^{N}\text{food}_{\text{c},i} \tag{38.3}$$

where food_{r} is the food returned by the end of the simulation and food_{c} is the food being carried by agents (but not yet returned) at the final time step. The summation is carried out over all N agents in the simulation.

At each time step, every agent will sense its local neighborhood and apply its behavioral algorithm. If an agent finds a patch containing food, it picks up and begins carrying five units of food. It carries the food to the home base, it drops the food, and the food then counts toward the system's total fitness. This is repeated for 1000 time steps, and the systems are judged at the end according to Eq. (38.3).

38.2.4 Scalability Assessment

To test for scalability, behavioral parameters are optimized and tested for each of one to six rows of agents. Then, keeping the parameters constant, the systems are tested in other scenarios. In this way, each system is tested both within and outside of the nominal size for which it was optimized. If increasing size leads to improved performance, we can call the system scalable. Because self-organizing systems have low integration costs, the majority of the cost of increasing system size comes from the actual agent hardware. If the performance of the system increases proportionally to the cost, we can call it linearly scalable. System scalability can also be superlinear if the performance increases with the number of agents and the rate of increase also increases. Sublinear systems get diminishing returns from adding agents, and in the worst case, their performance may even deteriorate.

38.2.5 Extended Optimization

The investigation in this paper requires that an optimal candidate be found for each scenario. In a complex system, it is very difficult to objectively determine which solution set is optimal for several reasons. The search space is enormous, and the simulations are partially stochastic, so to get statistical confidence, many trials would have to be performed. The GA is also partially stochastic, and different GA runs may converge to different parameter sets even if they are optimizing within the same search space. So in this paper, we do not refer to optimal candidates, but instead optimized candidates.

A consistent process was used to generate optimized candidates in each scenario, enabling a fair comparison of performance across scenarios. The process includes five GA runs. The three best candidates found from each of the first four runs are used to seed the fifth. The 15 best candidates (3 from each run) are then retested for reliability by evaluating their fitness in 100 simulations. The optimized candidate is chosen as the one with the highest 30th percentile performance out of these 100 runs.

38.3 Results and Implications

Note: the results show a distinction between systems *optimized for* a certain size and systems *deployed at* a certain size. In text, *RO-X* will denote the size a system is optimized for (where X represents the number of agent rows), and an *X-row* system will refer to the actual number of rows in deployment. For example, an RO2 six-row system refers to a system deployed with six rows of agents whose behavior was optimized for two rows of agents (Figs. 38.2, 38.3, 38.4, and 38.5).

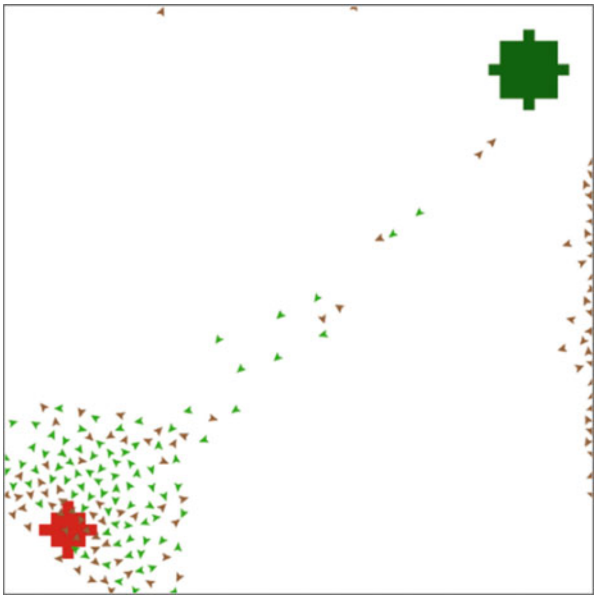


Fig. 38.2 RO1 six-row system showing jamming around home base

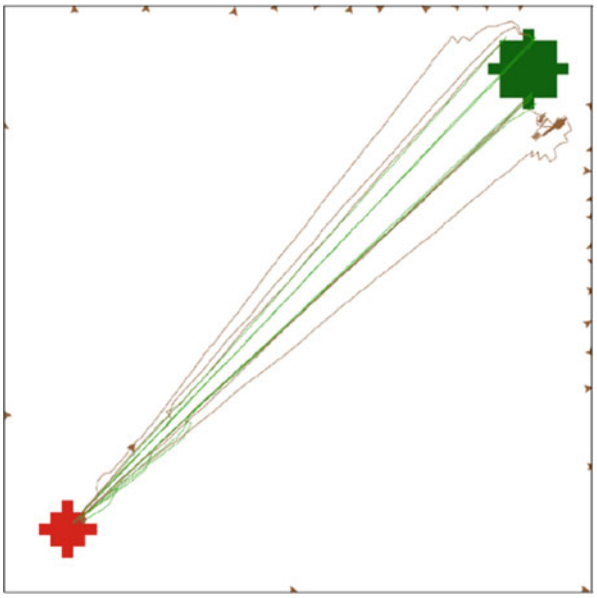


Fig. 38.3 Behavior of RO6 one-row system for time steps 500–1000

38.3.1 Scalability of Conceptual Design

Figure 38.5 shows the fitness of the optimized systems for each of one-to-six-row systems, after optimizing the parameters of Table 38.1 at each size. This

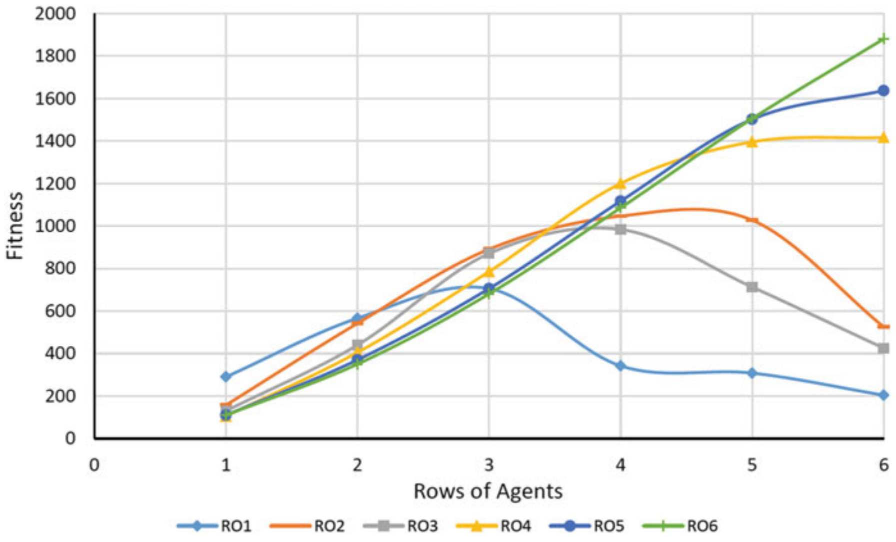


Fig. 38.4 Results of scalability test, where each curve represents one set of behavioral parameters, tested at each system size

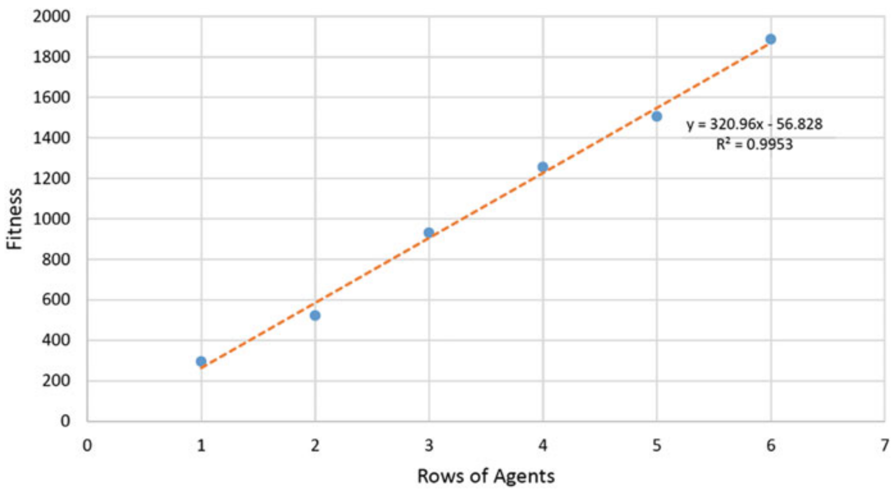


Fig. 38.5 Optimized fitness for each number of agent rows, showing a linear relationship between performance and system size

corresponds to the case where the designer can change behavioral parameters (i.e., change the detail design while maintaining the conceptual design) as more agents are added to the system. As seen in the figure, the system is almost perfectly linearly scalable. The R^2 value for the linear regression is 0.995.

38.3.2 Scalability of Detail Design

What can be done if the system operator is not allowed to change behavioral design parameters after design or deployment? In such a scenario, a system with parameters optimized for one row of agents may be scaled to six rows or vice versa. This constraint was explored by taking the optimized parameter sets from Sect. 38.3.1 and testing them for each of one to six rows of agents. Thus they were sometimes tested outside of the size range for which they were optimized. The results are summarized in Fig. 38.4.

It can be seen that systems optimized for small sizes (RO1, RO2, RO3) were unable to effectively scale up in size, but the systems optimized for large sizes were able to smoothly scale down in size. This implies that there is some directionality in the conceptual design, making it easier to decrease in size than to increase if the individual agent behavior is held constant. At either end, there was a fitness penalty for deploying a system at a given size larger or smaller than its optimized size, when compared with a system optimized for that size. (Note that due to small deficiencies in the partially stochastic optimization, there were several cases where this is not true. The most notable is two-row systems, where RO1 outperforms RO2 by 4.4%. All other cases manifest by a slimmer margin or have the expected system performing the best.) Table 38.2 shows the fitness penalties for cross-testing the extreme ends of the size range.

The most drastic fitness penalty comes from scaling the RO1 system to a six-row system. Figure 38.2 shows the end state of this scenario: the agents carrying food crowded around the home base and jammed the system. The reversed scenario (RO6 one row) also showed interesting results with a high fitness penalty. The strategy chosen by the GA during optimization (for a six-row system) was for the system to leave a layer of agents stuck along the boundary of the field, guiding a circulating inner core to find and retrieve food. This behavior is shown in Fig. 38.6. Contrast that figure with Fig. 38.3, which shows the RO6 one-row system attempting to exploit the same strategy, but it leaves too high a fraction of its agents static along the wall. With so few agents doing the foraging required to raise its fitness score, it suffers a 62% fitness penalty compared to the RO1 system.

Table 38.2 Results of cross-testing systems optimized for large size in small-scale deployment and vice versa

System size (rows)	Optimized fitness	Test system	Test fitness	Penalty (%)
1	290.0	RO5	110.0	62.1
		RO6	110.2	62.0
2	542.1	RO5	370.3	31.7
		RO6	350.3	35.4
5	1502.5	ROl	308.0	79.5
		RO2	1027.6	31.6
6	1881.1	ROl	204.0	89.2
		RO2	525.2	72.1

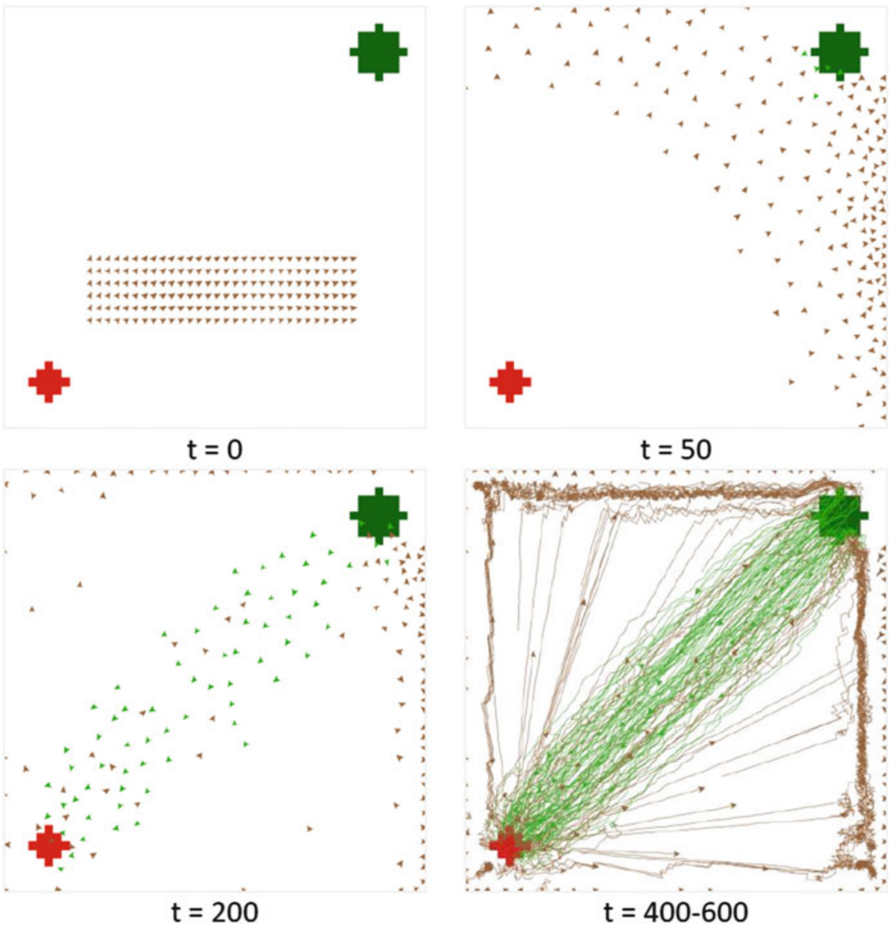


Fig. 38.6 Behavior of RO6 six-row system. The fourth panel shows a motion trace of each agent for 200 time steps

38.3.3 Scalability of System with Boundary Detection

It was shown in [22] that when agents are able to detect and react to the boundary of the field, they can flock and move much more smoothly, returning food more efficiently. This adds two behavioral parameters to the set from Table 38.1. A similar test was carried out for systems with boundary detection, and the results are shown in Fig. 38.7 and Table 38.3. The results follow the pattern of the previous section but are more extreme. There were fitness penalties up to 99% when scaling a system up in size and milder penalties when decreasing system size. The conceptual design was again linearly scalable (R2 = 0.999).

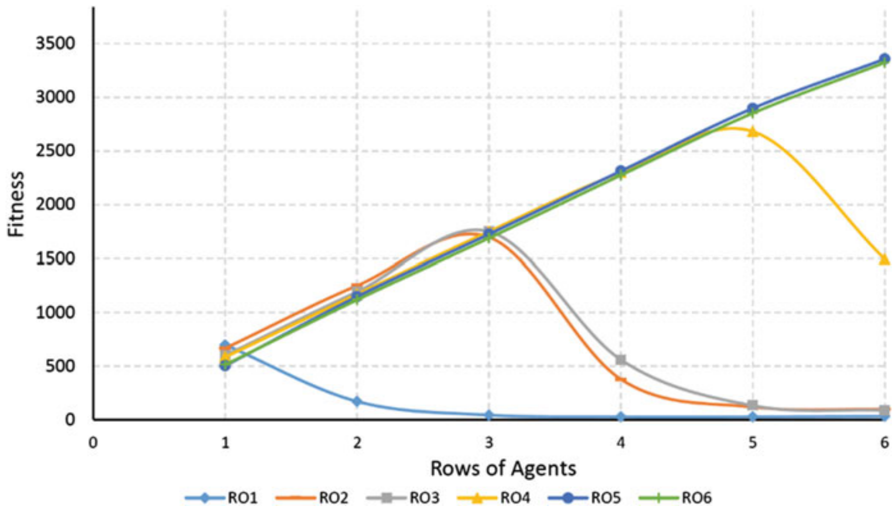


Fig. 38.7 Scalability tests for systems with boundary detection

Table 38.3 Results of cross-testing systems with boundary detection optimized for large size in small-scale deployment and vice versa

System size (rows)	Optimized fitness	Test system	Test fitness	Penalty (%)
1	697.4	RO5	502.3	28.0
		RO6	511.8	26.6
2	1249.1	RO5	1147.0	8.17
		RO6	1117.5	10.5
5	2896.7	RO1	25.75	99.1
		RO2	117.5	95.9
6	3323.3	RO1	30.45	99.1
		RO2	97,4	97.1

38.4 Discussion

At the conceptual design level, both sets of experiments showed linear scalability. For systems engineers, this means that scalable foraging systems are feasible, as long as there is a way to adjust the parameters according to the system size. Since the behavior is primarily dominated by software and parameters, this should not incur major cost.

However, the results show that it is difficult to scale a self-organizing system up in size if the behavioral parameters cannot be adjusted accordingly. Not only did the RO1 and RO2 systems suffer a relative fitness penalty as five-row and six-row systems compared to the RO5 and RO6 systems, but above a certain size, they even lose fitness in absolute terms. Looking at the simulation results qualitatively, agent groups cause jamming around the food or home base, halting progress early in the simulation run.

Scaling the RO5 and RO6 systems down in size was seen to be less problematic, although the relative fitness penalties were as high as 62%. The problem was that the behaviors and structures selected by the GA were quite effective at large sizes, but did not have the critical mass to be effective at small sizes. For example, in the RO6 six-row system without boundary detection, a small fraction of the agents could be sacrificed along the boundary in order to guide the rest of the agents toward the food, but as a one-row system, the number of static agents required for this strategy caused too much relative overhead and seriously limited the speed at which the system could find and return food.

38.5 Conclusion

38.5.1 Summary and Conclusions

With increasing connectivity and miniaturization of robots, there is greater opportunity for engineers to design distributed systems with self-organized architectures. The advantages of these systems are redundancy, adaptability, mass production, and possible scalability. Scalability was shown in this paper to be dependent on behavior parameters chosen during optimization. Systems optimized for small size suffered from jamming at large sizes, and systems optimized for large sizes did not have the resources to form large-scale subsystems at small sizes. The methodology, based on agent-based simulation, parametric behavioral modeling, and genetic optimization, was shown to effectively uncover these strategies and possible pitfalls. The engineer's responsibility is to leverage this information, knowledge of the system's environment, and use cases to tailor the agents' behavior to the appropriate size or size range.

38.5.2 Limitations and Future Work

The results of this paper are specific to the foraging simulations in question. They are meant to serve as data points in the study of self-organizing systems, and care must be taken in transferring specific results (e.g., fitness penalties) to the design of scalable complex systems in other domains. Nonetheless, the qualitative lessons, scalability assessment, and design methodology are generalizable to the design of many different self-organizing systems (see [21] for related examples). The simulation/optimization was also limited by the computational power available, as an exhaustive search of the 18-parameter space was impractical. With increased computational ability, the optimized systems will be closer to the true optimum.

For future work, we will look into various methods for adapting the behavior of the system to its specific size. This raises interesting questions of self-knowledge (how does a distributed system know its own size?) and how to distribute behavioral updates (should a central controller broadcast updates, or should they spread virally?). There are ongoing efforts to transfer the behavioral models in this research to physical robots, instead of just in simulation. Also, the general lessons on self-organization continue to inform our ongoing research on groups of autonomous ground, sea, and air vehicles [30, 31].

References

1. Ross AM, Rhodes DH, Hastings DE (2008) Defining changeability: reconciling flexibility, adaptability, scalability, modifiability, and robustness for maintaining system lifecycle value. Syst Eng 11(3):246–262
2. Buhl J et al (2006) From disorder to order in marching locusts. Science 312(5778):1402–1406
3. Humann J, Khani N, Jin Y (2016) Adaptability tradeoffs in the design of self-organizing systems. In: Proceedings of the ASME 2016 IDETC/CIE Conference, Charlotte
4. Matni N, Leong YP, Wang YS, You S, Horowitz MB, Doyle JC (2014) Resilience in large scale distributed systems. Procedia Comput Sci 28:285–293
5. Wischhof L, Ebner A, Rohling H (2005) Information dissemination in self-organizing intervehicle networks. IEEE Trans Intell Transp Syst 6(1):90–101
6. Dorigo M et al (2004) Evolving self-organizing behaviors for a swarm-bot. Auton Robot 17 (2–3):223–245
7. Beckers R, Holl OE, Deneubourg JL, Bielefeld Z, Bielefeld D (1994) From local actions to global tasks: stigmergy and collective robotics. In: Artificial Life IV. MIT Press, Cambridge, MA, pp 181–189
8. Petroski H (1994) Design paradigms: case histories of error and judgment in engineering. Cambridge University Press, Cambridge/New York
9. Bauza R, Gozálvez J (2013) Traffic congestion detection in large-scale scenarios using vehicle-to-vehicle communications. J Netw Comput Appl 36(5):1295–1307
10. Mikhailov AS (2011) From Swarms to Societies: Origins of Social Organization. In: Meyer-Ortmanns H, Thurner S (eds) Principles of Evolution. Springer Berlin Heidelberg, Berlin/Heidelberg, pp 367–380
11. Nowak DJ, Lamont GB, Peterson GL (2008) Emergent architecture in self organized swarm systems for military applications In: Proceedings of the 2008 GECCO conference companion on genetic and evolutionary computation, New York, pp 1913–1920
12. Reynolds CW (1987) Flocks, herds, and schools: a distributed behavioral model. In: ACM SIGGRAPH'87 conference proceedings, vol 25–34. Anaheim, CA
13. Song Y, Kim J-H, Shell D (2012) Self-organized clustering of square objects by multiple robots. In: Dorigo M, Birattari M, Blum C, Christensen A, Engelbrecht A, Groß R, Stützle T (eds) Swarm intelligence, vol 7461. Springer, Berlin/Heidelberg, pp 308–315
14. Zouein G, Chen C, Jin Y (2010) Create adaptive systems through 'DNA' guided cellular formation. In: Design creativity 2010. Springer, p 149
15. Jin Y, Chen C (2014) Cellular self-organizing systems: a field-based behavior regulation approach. Artif Intell Eng Des Anal Manuf 28(2):115–128
16. Chiang W, Jin Y (2011) Toward a meta-model of behavioral interaction for designing complex adaptive systems In: ASME IDETC/CIE 2011, pp 1077–1088
17. Khani N, Humann J, Jin Y (2016) Effect of social structuring in self-organizing systems. J Mech Des 138(4):041101

18. Humann J, Jin Y (2013) Evolutionary design of cellular self-organizing systems. In: ASME 2013 international design engineering technical conferences and computers and information in engineering conference, pp V03AT03A046–V03AT03A046
19. Jin Y, Chen C (2012) Field based behavior regulation for self-organization in cellular systems. In: Presented at the Design Computing and Cognition Conference DCC'12, 2012
20. Humann J, Madni AM (2014) Integrated agent-based modeling and optimization in complex systems analysis. Procedia Comput Sci 28:818–827
21. Humann J (2015) Computational synthesis and behavioral modeling of self-organizing systems. PhD, University of Southern California, Los Angeles
22. Humann J, Khani N, Jin Y (2014) Evolutionary computational synthesis of self-organizing systems. AI EDAM 28(03):259–275
23. Calvez B, Hutzler G (2006) Automatic tuning of agent-based models using genetic algorithms. In: Sichman J, Antunes L (eds) Multi-agent-based simulation VI, vol 3891. Springer, Berlin/ Heidelberg, pp 41–57
24. Holland JH (1992) Adaptation in natural and artificial systems : an introductory analysis with applications to biology, control, and artificial intelligence. MIT Press, Cambridge, MA
25. Goldberg DE (1989) Genetic algorithms in search, optimization, and machine learning, 1st edn. Addison-Wesley Professional, Boston
26. Camazine S (2001) Self-organization in biological systems. Princeton University Press, Princeton
27. Theraulaz G, Gautrais J, Camazine S, Deneubourg J-L (2003) The formation of spatial patterns in social insects: from simple behaviours to complex structures. Philos Trans R Soc London A: Math Phys Eng Sci 361(1807):1263–1282
28. Dorigo M, Blum C (2005) Ant colony optimization theory: a survey. Theor Comput Sci 344 (2–3):243–278
29. Wilensky U (1998) NetLogo. Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston
30. Ordoukhanian E, Madni AM (2017) Introducing Resilience into Multi-UAV SoS. In: Presented at the 15th annual Conference on Systems Engineering Research (CSER), Redondo Beach
31. Madni AM, Sievers MW, Humann J, Ordoukhanian E, D'Ambrosio J, Sundaram P (2017) Model-based approach for engineering resilient system-of-systems: application to autonomous vehicle network. In: Presented at the 15th annual Conference on Systems Engineering Research (CSER), Redondo Beach

Chapter 39
Evaluation of Cross-Project Multitasking in Software Projects

Alexey Tregubov, Jo Ann Lane, and Barry Boehm

Abstract It has been observed that multitasking can cause inefficient (or unproductive) work. Modern lean and agile practices in software engineering processes also acknowledge the problem and attempt to eliminate waste by limiting work in progress and using better team organization and work scheduling techniques. Existing research has studied multitasking and work interruptions on individuals, but very few of them have evaluated the effects of multitasking on the team or the whole organization. The goal of this study is to understand how multitasking and interruptions affect the cost of software projects. In this paper, we present a method for quantitative evaluation of the negative impact of cross-project multitasking in software development. The presented method can serve as a tool for better effort estimation as well as a metric for productivity evaluation in multiproject environments. The method was used to evaluate cross-project multitasking overhead of several industry projects. Additionally, we studied whether the number of projects in which developers were involved simultaneously is a predictor of the number of work interruptions and multitasking overhead in the team.

Keywords Multitasking in software engineering • Effort estimation • Multitasking evaluation • Multitasking overhead • Cost and schedule estimation

39.1 Introduction

Modern business often relies on multitasking to optimize resource utilization. However, according to studies in [1–5], multitasking is one of the causes of unproductive work. To deal with negative effects of multitasking, various project management frameworks (e.g. Agile, Scrum, Kanban) developed different practices.

There are also benefits of limited multitasking and interruptions. For example, research [3] suggests that if one is working on a routine task, "multitasking breaks the boredom," reducing the inclination to procrastinate and providing a stimulating

A. Tregubov (✉) • J.A. Lane • B. Boehm
University of Southern California, Los Angeles, CA, USA
e-mail: tregubov@usc.edu; jolane@usc.edu; boehm@usc.edu

A.M. Madni et al. (eds.), *Disciplinary Convergence in Systems Engineering Research*, https://doi.org/10.1007/978-3-319-62217-0_39

environment. The same applies to meetings and other collaboration and coordination activities. For instance, design meetings that help developers to understand the emerging "big picture" and peer reviews for quality and sharing knowledge are necessary for any project. A lack of such communication can cause quality to degrade, rework to increase, and slow down the whole workflow.

To accelerate development schedules and optimize resource utilization, software development organizations may choose working on several projects in parallel. Depending on team and organization structure, it may lead to situations where scarce resources are shared among several projects at a time [6, 7]. While concurrent work on several projects can accelerate development schedules via better resource utilization, there is a downside – resource multitasking introduces additional overhead, which essentially affects overall productivity.

In this paper, we define cross-project multitasking as an involvement of individuals in multiple engineering activities with different contexts of work over a certain period. Different projects and different versions or releases are the most common examples of different work contexts. Switching between different work contexts can affect resources' productivity.

Cross-project multitasking may appear in different forms. For example:

- In organizations with matrix structure, resources are shared between several projects by design for better resource utilization [8, 9].
- In projects, where multiple releases of the product are maintained, resources are shared between maintenance of the previous releases and new versions.
- In system-of-systems environments, if a constituent system is developed for several customers (e.g. different software distributions/releases for each customer), resources are shared between different contexts. The context here is customer-specific requirements, success-critical stakeholders, and everything that makes each system installation unique [9–12].

The goal of the research is to understand how multitasking and work interruptions can affect the cost of software projects.

In this research, we evaluate the negative impact of cross-project multitasking overhead in software development projects. We also present an approach for quantitative evaluation of the negative impact of cross-project multitasking in software development projects. The presented method can serve as a tool for better effort and schedule estimation as well as a metric for productivity evaluation. The method was applied to evaluate the impact on productivity in several industry projects.

39.2 Background

39.2.1 Multitasking and Productivity

Multitasking and work interruptions have become an inevitable part of the work process of knowledge workers. It has been observed that the structure of work time has become less predictable and certain [13]. For the most part, the work process is not a list of activities performed sequentially, it is a mixture of long-term goals (e.g. project deadlines, milestones) and short-term tasks (e.g. meetings, negotiations). In one of the case studies, O'Carroll observed that the "key" activities of software engineers (e.g. thinking on a problem, designing, programming, etc.) are often interrupted with collaboration activities (meetings, emails, chatting with colleagues) [13]. Although it may not be always perceived as an "actual work," these collaboration activities are an essential part of work allowing information exchange. One of the drawbacks of such work environments is that workers often have to multitask, for instance, interrupt their work, which may affect their productivity of doing the "actual work."

Various project management frameworks developed different practices to limit the negative impact of interruptions and multitasking. For example, many agile-based methodologies propose a role of a process facilitator. The process facilitator not only guides the team to follow the process but also serves as a buffer between the team and external distractions. Kanban-based processes explicitly limit work in progress and reduce switching between tasks. Scrum methodology explicitly regulates meetings as well as their length, time, and frequency to prevent unnecessary work interruptions and waste of time.

Not all multitasking and interruptions at work are necessarily inefficient. First, limited multitasking and interruptions serve as a mechanism for information exchange. Second, in some situations, interruptions may actually increase productivity. For example, people working on a large project often need to solve complex problems requiring a lot of cognitive involvement. At some point, we may find that one is struggling too much with a problem. It often helps to go work on something else and let the subconscious mind work on the problem for a little while. In this case, switching to another task may reduce "struggle time." This observation was noted in [14, 15] as a way to increase the productivity of complex creative work.

39.2.2 Definition of Multitasking and Cost of Switching Between Tasks

Multitasking is a term usually used to describe the activity of performing multiple tasks during a certain period of time [1]. It is defined as the "engagement in individual and discrete tasks that are performed in succession." It is implied that

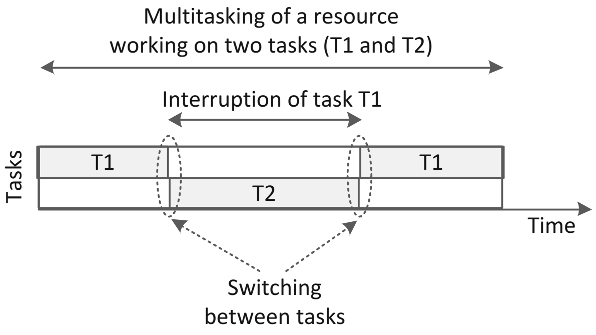


Fig. 39.1 Multitasking defined as tasks that are performed in succession

there is switching between tasks in succession [3]. This definition of multitasking also establishes a relationship between task switching and interruptions (Fig. 39.1).

In this paper, we do not discuss multitasking without interruptions and switching between activities because usually this kind of multitasking is related to simple mechanical activities, and it does not require significant cognitive attention (e.g. chewing and walking, listening to music, typing, etc.).

Neuroscience researchers [1] conducted task-switching experiments in an effort to measure the "cost" or loss of time spent switching between activities. Additionally, it reduces the amount of short-term memory dedicated to each of the tasks, potentially making an individual less productive. In general, it all depends on the type of tasks (e.g. complex tasks vs. simple tasks) and the type of interruptions (e.g. long interruptions vs. short interruptions).

The Microsoft research [16] studied 11 information workers (software developers, web designers, network administrators, etc.) in a weeklong study where they recorded the frequency of interruptions of the performed tasks. All participants were working on several projects (tasks distribution across projects was not recorded). Their research identified that 27% of all tasks were routine or project tasks ("actual" work), and the rest of the tasks are communication/collaboration tasks. This finding is consistent with a study of DeMarco and Lister [17]. DeMarco and Lister observed that software developers work only 30% of their time alone, while the rest of their time is dedicated to collaborative work where several people are involved. The Microsoft research [16] also finds that 26% of all interruptions were caused either by new or returning tasks, and 40% of all interrupted tasks were not resumed immediately (long interruptions). This body of research suggests the difficulty that workers experience with returning to interrupted tasks.

DeMarco and Lister studied multitasking and attention switching from a different perspective. To measure the effect of attention switching, they measured number of uninterrupted hours and body-present hours [17]. They defined an E-factor (39.1):

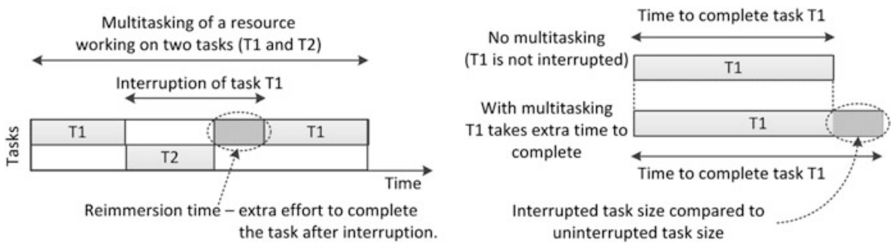

Multitasking of a resource working on two tasks (T1 and T2)
Interruption of task T1
Tasks
T1
T2
T1
Time
Reimmersion time – extra effort to complete the task after interruption.
Time to complete task T1
No multitasking (T1 is not interrupted)
T1
With multitasking T1 takes extra time to complete
T1
Time to complete task T1
Interrupted task size compared to uninterrupted task size

Fig. 39.2 Reimmersion time

$$E\text{−factor} = \frac{\text{Uninterrupted hours}}{\text{Body present hours}} \tag{39.1}$$

In some cases, they recorded a range of E-factor in their experiments from 0.10 to 0.38. This could serve as an approximation of attention switching effects on productivity.

To measure the lasting effect of interruptions, DeMarco and Lister introduced a concept of the “reimmersion time” [17] (Fig. 39.2). They note that “If average incoming phone call takes five minutes and your reimmersion period is fifteen minutes, the total cost of that call in flow time (work time) lost is twenty minutes. A dozen phone calls use up a half day.”

Weinberg [2] suggests the following heuristic (Table 39.1) when estimating effects of multitasking on an individual. The heuristic estimates cost of multitasking as a linear function of a number of tasks being done in parallel. It estimates that every additional task/project increases effort spent on context switching by 20%.

In general, multitasking may have positive/neutral and negative effects on workers’ productivity. In this paper, we only studied a negative impact of cross-project multitasking.

39.3 Cross-Project Multitasking in Software Projects

39.3.1 Research Questions

We can find various causes for interruptions in our daily work (Fig. 39.3). For example, individuals’ working and learning styles (e.g. closure-oriented vs. open-oriented) affect the reimmersion time and how often a person tends to jump from one task to another. Different work environments and external influences (e.g. requirements volatility) can introduce coordination interruptions. Additionally, work interruptions not only may cause destructions but also can bring new job insights.

In this paper, we discuss only interruptions caused by cross-project multitasking and their negative impact on teams’ productivity (Fig. 39.3). Cross-project

Table 39.1 Weinberg's heuristic

Number of tasks	% of Time on each	% of Time on switching between tasks
1	100	0
2	40	20
3	20	40
4	10	60
5	5	75
6 and more	Random	Random

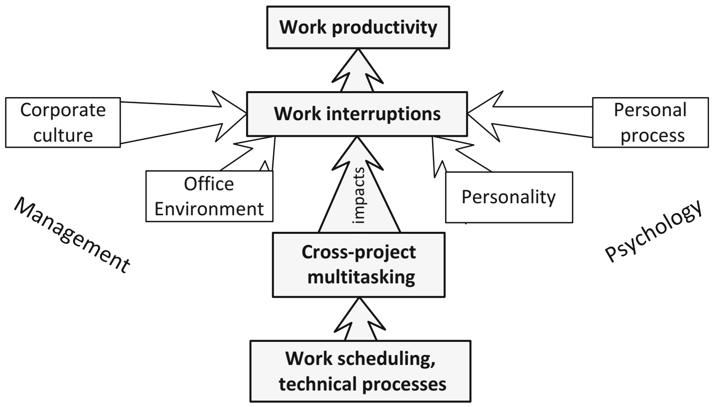


Fig. 39.3 Causes for interruptions

multitasking may constitute a significant portion of interruptions in daily work [16]. These interruptions tend to introduce costly context switching between projects. The impact of each such interruption is sometimes estimated with hours of time loss [17, 18].

The following research questions are addressed in the analysis, performed on the collected data:

- Is the cross-project multitasking overhead linearly correlated with the number of projects?
- What is the quantitative effect of cross-project multitasking on development effort?

Additionally, we also compared the results with the Weinberg's heuristic (Table 39.1).

Understanding of the overall impact of interruptions on software projects can be used to improve organization's processes and effort estimates of the future development. Cross-project multitasking can often be explained by work schedule and the organization's workflow. An excessive cross-project multitasking can be an indicator of improper resource/skills distribution between projects.

39.3.2 Data Collection

To evaluate the impact of cross-project multitasking, we collected work logs of 81 software developers for 1 year of work on six projects. All observed projects were completed within 1 year. All software developers worked in one company. The company had a matrix organizational structure, which allowed excessive resource sharing across multiple projects. The company's primary domain of work is the development of large-scale distributed systems for electrical energy consumption monitoring in large cities. The company was a private software development organization, which worked on software products for smart grid solutions.

Daily work logs were part of the organization's work process, which allowed us to collect effort data and its distribution across projects and individuals without interfering with their workflow. Project tasks were assigned to developers via the project tracking system used in the company. Every developer reported their daily progress (time spent) on each task he/she worked on that day. Depending on schedule and urgent requests, developers could work on tasks from different projects in 1 day. The project tracking system also tracked tasks' statuses. A task's status, reported by developers, can show if it is completed, in progress, or in the backlog. In this study, we analyzed only work logs of software engineers, so all the tasks reported in work logs were mostly development tasks (architecting, designing, coding, bug fixing, testing, documenting the source code, etc.).

39.3.3 Work Log Analysis

Work interruptions were counted by analyzing developers' work logs. If a developer worked on more than one project in 1 day, it meant that he had to multitask between them. Using tasks' statuses, we excluded cases when one task was completed and another one started in 1 day (no multitasking in such cases). Then we evaluated the number of interruptions of work. Without interviewing the developers, there is no way to know the exact number of work interruptions they experienced switching between tasks in 1 day. However, we can count the minimum possible number of interruptions they had. For example, if a developer worked on two tasks in 1 day and continued working on them on the next day, he experienced at least one interruption on the first day. We applied this logic to count interruptions that developers experienced switching between tasks from different projects.

We wanted to be focused on the interruptions that cause significant context switching and require extra attention and cognitive effort; for that reason, we only counted tasks from different projects. Projects had relatively different contexts: different software products from the same product line, maintenance (defect fixing) of different releases. Switching between them required not only time to rebuild a mental model of the task but also time to switch between different databases, source code repositories, and development environments. It could take at least 20–90 min to do so.

We used the reimmersion time [8] (Fig. 39.2) to model the impact of interruptions on developers' productivity. This approach allows us to evaluate the impact of multitasking on the effort. To apply it, we need to know the number of interruptions and the reimmersion time of each interruption. We counted the number of tasks interruptions as described above. The reimmersion time may depend on many factors: the complexity of the task, length of the interruption, complexity of the other task that interrupted work, etc. Different sources estimate that the reimmersion time for information/knowledge workers varies from 20 min to 1–2 h. In this study, we assumed a constant reimmersion time for all the interruptions we counted. Such assumption is justified because we only counted development tasks (work of a similar nature) and only interruptions that required a different context of work (a different project). In other words, conditions for the reimmersion time were relatively similar, and this allows us to assume that the reimmersion time was also relatively the similar across all interruptions.

The estimated value for the reimmersion time is a parameter of the model. To evaluate the impact on the effort, we used reimmersion time values in the range of 20–120 min.

For each project, we evaluated multitasking overhead as follows (39.2):

$$\text{Multitasking overhead} = \frac{\text{Total number of interruptions} \times \text{Reimmersion time}}{\text{Total effort}} \tag{39.2}$$

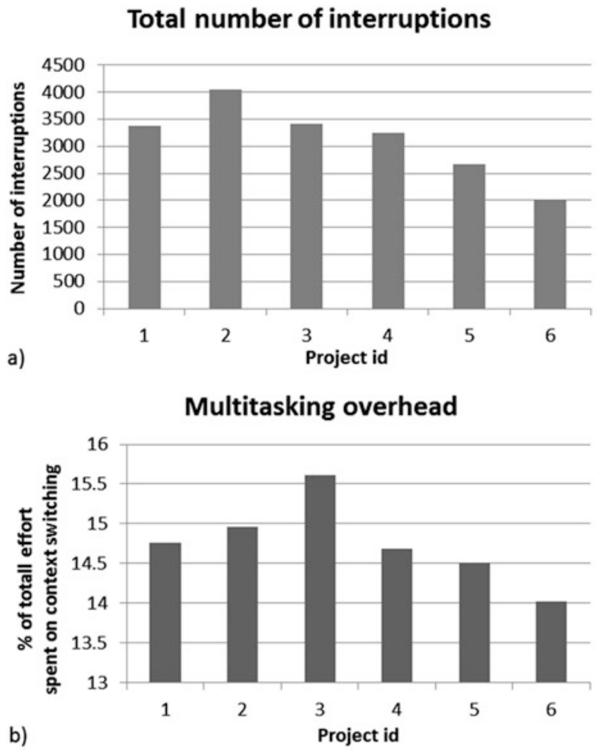
39.4 Results

Figure 39.4 shows (a) total number of interruptions and (b) multitasking overhead, where the reimmersion time was estimated as 1 h. The lower bound of the cross-project multitasking overhead was estimated between 14.02% (for project 6) and 15.61% (for project 3) of the overall effort spent in a project.

Not all projects were equally affected by cross-project interruptions. Figure 39.5 shows the average number of interruptions per week for each project.

For each week, we counted how many cross-project interruptions each developer experienced as well as time spent by each developer. We also counted on how many different projects each person worked in each week. Figure 39.6 shows (a) the number of interruptions in each week (~40 of them) for each developer versus the number of projects in which each developer was involved in each week and (b) the average number of interruptions per week for each developer versus the average number of projects per week each developer worked on.

Fig. 39.4 (**a**) The total number of interruptions in each project. (**b**) Multitasking overhead



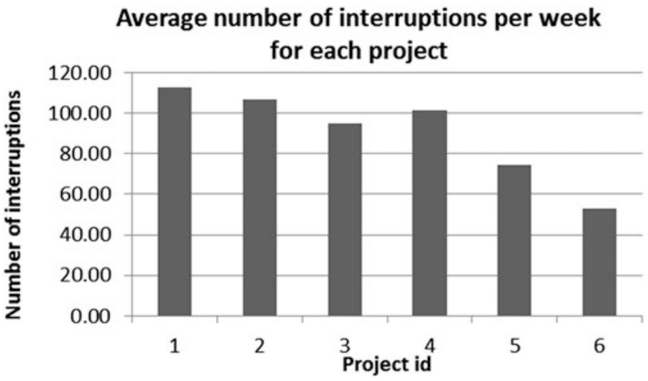


Fig. 39.5 The average number of interruptions per week in each project

Each data point in Fig. 39.6a is a number of interruptions and effort of one person in 1 week. The total number of data points is 3073. Weeks when developers did not work (0 time spent) were excluded from the data set. Each data point in Fig. 39.6b is an average number of interruptions per week and average effort (averaged across all weeks) of one developer. The total number of data points is 81.

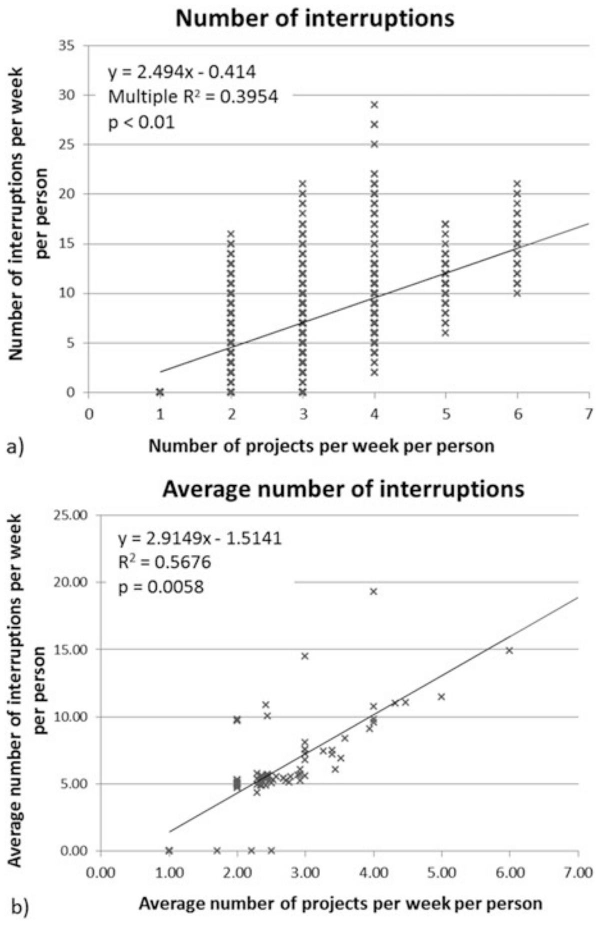


Fig. 39.6 (**a**) The number of interruptions per person per week versus the number of projects in which the person was involved. (**b**) The average number of interruptions per week per person versus the average number of projects per week each person worked on

We computed the number of interruptions in each week of each developer over the course of a year; therefore, we had to account for repeated observations in our linear regression analysis.

To see if the increase of the number of projects a developer worked on is associated with increase of the number of interruptions and effort of the developer (a correlation within subjects), we used a multiple linear regression analysis to compute correlation coefficients. We treated developers as a categorical factor (a predictor variable) [19] (Fig. 39.6a). The linear correlation in this analysis is weak multiple $R^2 = 0.395$. This finding suggests that a number of projects, in which a developer is involved, might not be a good predictor for work interruptions he/she is experienced. In other words, a developer working on two to three projects can experience a similar amount of work interruptions per week as a developer working on four to five projects.

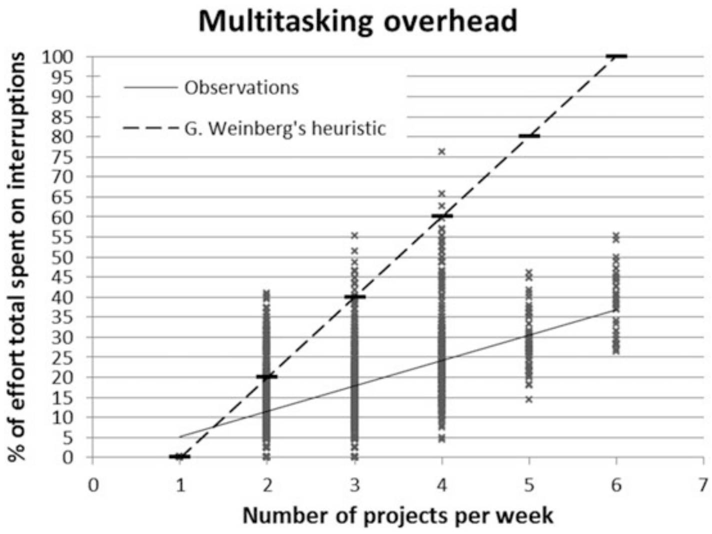


Fig. 39.7 Comparison of Weinberg's heuristic with observations (lower bound estimate for multitasking overhead)

To see if developers working on more projects tend to have more interruptions and tend to spend more effort on multitasking overhead, we did a linear regression analysis using mean values [20] (a correlation between subject means in Fig. 39.6b). The linear correlation in this analysis is characterized by $R^2 = 0.568$, which suggests that a team of developers multitasked between four and five projects experiences more interruptions than a team of developers working on two to three projects.

We compared multitasking overhead results with the Weinberg's heuristic (Fig. 39.7). Weinberg's heuristic predicts a larger multitasking overhead. As we only evaluated the lower bound of the cross-project multitasking overhead, the overall multitasking overhead, including multitasking within each project/task and across different projects, should be higher.

For each week, we also counted how many developers were involved in how many projects (Fig. 39.8). Figure 39.8 shows distribution averaged over a 38-week period. Most of the time, many developers (56%) worked on only three projects per week and less than 1% of them worked on all six projects.

39.5 Conclusions

In this study, we evaluated the impact of cross-project multitasking on software projects by analyzing work logs. We analyzed work logs of 81 software developers working on six projects for 1 year. The evaluation showed that among all six

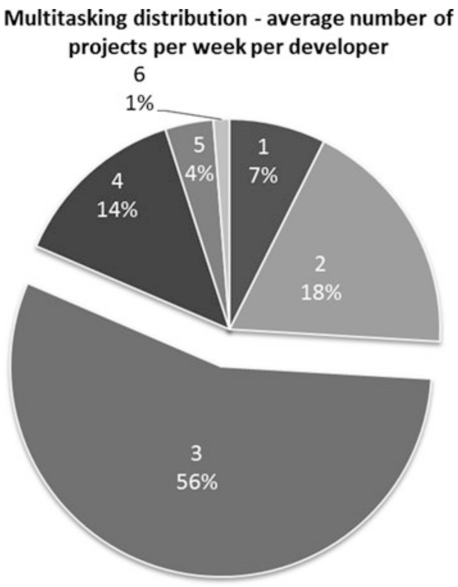


Fig. 39.8 Multitasking distribution – average number of projects per week

projects at least 14% effort was spent on context switching between tasks from different projects. Developers who were involved in more projects tend to have more cross-project work interruptions. However, the linear correlation between the number of projects each resource is working on in 1 week and the number of interruptions is relatively weak.

We identified the following threats to validity of this research:

- Work logs may contain inaccurate data. For example, it is possible that in some occasions work status was updated by a day later, which affects the number of counted interruptions.
- The choice of the reimmersion time values was based on literature search and experience of individual developers. Unfortunately, there was no opportunity to conduct a survey among all of the developers to better estimate the reimmersion time value.

To address these threats, we intend to study multitasking in other organizations as well.

As this study demonstrates, the number of interruptions can be used to evaluate the impact of multitasking on the effort. The number of interruptions that people experience can also be used as a metric and a measure of multitasking in teams.

39.6 Future Work

The impact of multitasking on effort can be integrated into parametric cost and schedule estimation models such as the Constructive Cost Estimation Model (COCOMO®) for better effort estimation.

The approach for counting work interruptions and evaluating their impact on effort can be applied to other types of multitasking. The work log analysis tools, we used in this research, can be integrated with project tracking systems such as Atlassian Jira to provide real-time information about work interruptions and their impact on productivity.

Acknowledgments This material is based upon work in part supported by the U.S. Department of Defense through the Systems Engineering Research Center (SERC) under contract H98230-08-D-0171 and documented in technical reports SERC-TR-022. This material is also based upon work supported, in whole or in part, by the U.S. Department of Defense through the SERC under contract HQ0034-13-D-0004. SERC is a federally funded University Affiliated Research Center managed by Stevens Institute of Technology.

References

1. Dzubak CM (2008) Multitasking: the good, the bad, and the unknown. J Assoc Tutor Prof 1 (2):1–12
2. Weinberg GM (1993) Quality software management. Dorset House, New York
3. Delbridge KA (2000) Individual differences in multi-tasking ability: exploring a nomological network. Unpublished Doctoral Dissertation, University of Michigan
4. American Psychological Association (2006) Multitasking: switching costs. http://www.apa.org/research/action/multitask.aspx
5. Tregubov A, Lane JA (2015) Simulation of Kanban-based scheduling for systems of systems: initial results. Procedia Comput Sci 44:224–233
6. Appelbaum SH, Marchionni A, Fernandez A (2008) The multi-tasking paradox: perceptions, problems and strategies. Manag Decis 46(9):1313–1325
7. Ford RC, Randolph WA (1992) Cross-functional structures: a review and integration of matrix organization and project management. J Manag 18(2):267–294
8. Galbraith JR (1971) Matrix organization designs how to combine functional and project forms. Bus Horiz 14(1):29–40
9. Brykczynski B, Stutz RD (2006) Software engineering project management. John Wiley & Sons
10. NDIA-National Defense Industrial Association (2010) Top Systems Engineering Issues In US Defense Industry. Systems Engineering Division Task Group Report. http://www.ndia.org/Divisions/Divisions/SystemsEngineering/Documents/Studies/Top%20SE%20Issues%202010%20Report%20v11%20FINAL.pdf
11. Turner R, Shull F, Boehm B, Carrigy A, Clarke L, Componation P, Dagli C, Lane JA, Layman L, Miller A, O'Brien S (2009) Evaluation of Systems Engineering Methods, Processes and Tools on Department of Defense and Intelligence Community Programs-Phase 2 (No. SERC-2009-TR-002). Systems Engineering Research Center, Hoboken
12. Turner R, Lane JA (2013) Goal-question-Kanban: applying lean concepts to coordinate multi-level systems engineering in large enterprises. Procedia Comput Sci 16:512–521

13. O'Carroll A (2015) Spaghetti time: organisational culture, multi-tasking and boundaries. In: Working time, knowledge work and post-industrial society. Palgrave Macmillan, London, pp 54–70
14. Leonard-Barton D, Swap WC (1999) When sparks fly: igniting creativity in groups. Harvard Business Press, Boston
15. Jett QR, George JM (2003) Work interrupted: a closer look at the role of interruptions in organizational life. Acad Manag Rev 28(3):494–507
16. Czerwinski M, Horvitz E, Wilhite S (2004) A diary study of task switching and interruptions. In: Proceedings of the SIGCHI conference on human factors in computing systems ACM, pp 175–182
17. DeMarco T, Lister T (2013) Peopleware: productive projects and teams. Addison-Wesley, Upper Saddle River
18. O'Carroll A (2008) Fuzzy holes and intangible time. Time in a knowledge industry. Time Soc 17(2–3):179–193
19. Bland JM, Altman DG (1995) Calculating correlation coefficients with repeated observations: part 2—correlation between subjects. BMJ 310(6980):633
20. Bland JM, Altman DG (1995) Statistics notes: calculating correlation coefficients with repeated observations: part 1—correlation within subjects. BMJ 310(6977):446

Chapter 40
Cultural Worldviews on an Aerospace Standards Committee: A Preliminary Analysis

J. John Park and David A. Broniatowski

Abstract Technical committees for industry consensus standards involve multiple stakeholders. These stakeholders are experts who assess and perceive benefits and risks differently due to differences in their experiences, training, and cultural worldviews. Decision-making on technical committees is premised on information sharing and communications between these experts. We seek to understand how these worldviews drive decision-making criteria. Specifically, we aim to test the hypothesis that technical experts' worldviews are diagnostic of their technical preferences. In this paper, we surveyed members of the National Aerospace Standards Committee (NASC). We report preliminary results in our efforts to develop a survey that can reliably measure their cultural worldviews. Our approach was inspired by measures used by Kahan's cultural cognition paradigm, a method of categorizing individuals' worldviews and associated risk perceptions on a combination of Douglas's cultural theory of risk, and Slovic's Psychometric Paradigm. Preliminary results indicate support for the existence of different worldviews on the NASC.

Keywords Cultural theory • Cultural cognition • Technical committee • Decision-making • Standard

40.1 Introduction

Technical committees for industry consensus standards necessarily involve multiple stakeholders who are experts with professional specialties, representing domains of knowledge. As they frequently disagree when making technical decisions, this paper explores why.

This paper proceeds from the premise that decision-making on technical committees requires information sharing and communications between experts who assess and perceive benefits and risks differently due to their possessing different worldviews [1, 2]. Broniatowski [3] has hypothesized that these worldviews are

J.J. Park (✉) • D.A. Broniatowski
The George Washington University, Washington, DC, USA
e-mail: pjpark@gwu.edu; broniatowski@gwu.edu

A.M. Madni et al. (eds.), *Disciplinary Convergence in Systems Engineering Research*, https://doi.org/10.1007/978-3-319-62217-0_40

also associated with the generic architectures of human organizations and technical systems, as well as specific approaches to system architecture. Furthermore, Moses [4] posits that the architecture of human organizations tends to be consistent with their cultural values such as emphasis on hierarchy and competition versus cooperation. He further posits that these architectures are associated with how information flows between system components [5]. Here, we speculate that the "cultural worldviews" of committee members are associated with training in a specific discipline, selection of goals, and the methodological preferences to achieve those goals. Thus, we seek to understand how committee members make decisions and the extent to which these might be based on their cultural worldviews.

To test these ideas, we aim to survey members of the National Aerospace Standards Committee (NASC) to test the hypothesis that technical experts' worldviews would be diagnostic of their technical preferences. In this paper, we report preliminary results in our efforts to develop a survey that can reliably measure these worldviews. Our approach was inspired by measures used by Kahan's Cultural Cognition project [6]. Specifically, we aimed to define technical analogs of the group and grid (solidarism-individualism; hierarchy-egalitarianism) axes that anthropologist has used to define cultural worldviews in the context of the cultural theory of risk [7, 8]. The cultural theory of risk posits four types of societies – "hierarchy," "sects," "markets," and "isolates" – each of which shapes how its members perceive and respond to risks from technology. Members of any of these societies will pay attention to those risks that are incompatible with their society's social structure. For example, hierarchical societies will pay attention to risks that challenge the basis of the hierarchy. An extensive review of the cultural theory of risk may be found in the book by Douglas and Wildavsky [1]. Our goal is to determine if these worldviews drive decision-making criteria and the risks perceived by NASC members.

40.2 Literature Review

40.2.1 Cultural Theory of Risk

The cultural theory of risk suggests that members of societies perceive risks associated with technologies based upon four types of societies defined by along two orthogonal dimensions (Fig. 40.1), known as "group" and "grid" [7].

The group dimension measures "the degree of social incorporation of the individuals in a social unit" [9]. An individual with a high group way of life tends to believe in solidaristic worldview, depending "on each other, which promotes values of solidarity rather than the competitiveness of weak group" [9]. A weak group way of life predisposes an individual to individualistic worldview, emphasizing fending for oneself and competition in lieu of cooperation [9]. The grid dimension defines a hierarchical-egalitarian scale, measuring whether the roles

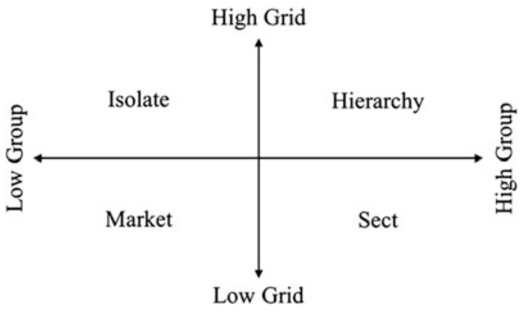


Fig. 40.1 The grid-group diagram (Adapted from Douglas [7, 8])

of society's members are stratified and differentiated. A high grid way of life is conducive to a "hierarchic" worldview in which the society determines individuals' social roles by their positions relative to external distinctions such as expertise and social classification. A low grid way of life, on the other hand, is conducive to "egalitarianism," expressing the belief that no one should be prevented from participating fully in any social role regardless of their status and classification in the societies [9].

The fundamental argument of the cultural theory of risk is that individuals perceive risks consistent with "cultural way of life" associated with an individual's underlying beliefs about the world [10]. Psychologically, individuals tend to believe that behavior they believe right is socially beneficial, and behavior they believe wrong is harmful [11].

40.2.2 Empirical Measures of the Cultural Theory of Risk

There are several techniques to measure the cultural worldviews hypothesized by the cultural theory of risk. Karl Dake's dissertation published in the early 1990s was the first empirical study of the cultural theory of risk [2]. His framework was designed to test cultural biases referring to shared beliefs and values, as well as contemporary worldviews based upon an individual's personality and political orientation [12]. He maintained that this collective approach is crucial to understand how the society's members perceive risks.

In doing so, Dake designed a survey with separate scales for hierarchy, egalitarianism, and individualism with the aim of distinguishing competing worldviews [2]. Peters and Slovic [13] built on Dake's work when examining cultural worldviews surrounding nuclear power. They found that the scales posited by Dake [2] were not statistically independent, leading Kahan [6] to critique the internal validity of Dake's work.

40.2.3 Cultural Cognition of Risk

Kahan interpreted Peters and Slovic's data to indicate that only two dimensions were necessary to appropriately capture the effect of cultural worldviews on risk perceptions. As such, he introduced the cultural cognition paradigm: a method of categorizing individuals' worldviews and associated risk perceptions based on a combination of cultural theory of risk [2] and Slovic's psychometric paradigm [14].

In his studies, Kahan adapted the items used in previous surveys of the cultural theory of risk including Dake's study and that of Peters and Slovic. Specifically, Kahan interpreted the results of these studies to indicate that existence of "two continuous attitudinal scales," one of which maps to the grid dimension (which Kahan interprets as hierarchy vs. egalitarianism) and the other for the group dimension (which Kahan interprets as individualism vs. solidarism), when measuring the worldviews of general population samples in the United States. Like the cultural theory of risk, Kahan's method assigns four ways of life – "hierarchical individualism," "hierarchical communitarianism," "egalitarian individualism," and "egalitarian communitarianism" – into the group-grid quadrants (Fig. 40.2) [6].

Unlike the cultural theory of risk, whose unit of analysis is the level of society, the cultural cognition hypothesis analyzes the level of the individual. Furthermore, individuals can have intermediate positions on one or both axes. That is, each individual's worldview is identified with a unique point in the two-dimensional space corresponding to the scales for hierarchy and individualism.

40.2.4 Critiques of the Cultural Cognition Hypothesis

Pointing out that the cognitive mechanisms proposed by the cultural cognition are specific for United States and draw mostly from a narrow class of findings in social psychology, van der Linden asserts that the cultural cognition has only been applied to issues with a strong political valence, and thus the theory tends to greatly exaggerate its generalizability [15]. We aim to determine if the cultural cognition paradigm is applicable to nonpolitical groups of technical experts.

40.3 Method

We designed a survey by adapting Kahan's cultural cognition survey items. We distributed our survey to 215 registered members of the NASC between April 6th and 22nd, 2016 using Qualtrics, an online survey software package. Survey responses were collected anonymously, and no incentives were offered. The protocol was approved by the George Washington University Institutional Review Board (Study No: 031658) and the Aerospace Industries Association of America.

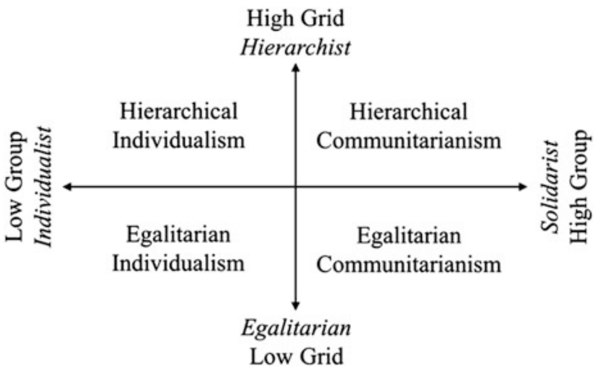

High Grid
Hierarchist
Hierarchical Individualism
Hierarchical Communitarianism
Low Group
Individualist
Solidarist
High Group
Egalitarian Individualism
Egalitarian Communitarianism
Egalitarian
Low Grid

Fig. 40.2 Four ways of life into group-grid space (Adapted from Kahan [6])

40.3.1 Subjects

Since 1938, the NASC has been responsible for developing and maintaining the NAS standards, and reviewing and commenting on similar government specifications and standards. The stakeholders in the NASC represent aircraft Original Equipment Manufacturer (OEMs), part manufacturers, part distributors, inspection laboratories, the government agencies, as well as a small number of individual researchers and consultants. The members of the NASC represent not only the organization that they work for but also technical expertise itself as individual experts of the industry.

40.3.2 Cultural Worldview Measures

To determine if we could categorize members of technical expert committees into cultural worldviews that are relevant to the cultural theory of risk, we adapted Kahan's cultural cognition survey items [6] to this research (Table 40.1). Specifically, we reinterpreted Kahan's survey items to be suitable for measuring cultural worldviews of individual members of the committee. We defined two continuous attitudinal scales based on Kahan's work, namely "hierarchy-egalitarianism" and "individualism-communitarianism" as representative of grid and group axes, respectively [6].

We translated several of Kahan's items into terms that we posited would be relevant to NASC members, while still retaining the flavor of the cultural theory of risk. For example, "A lot of problems in our society today come from the decline in the traditional family, where the man works and the woman stays home" in Kahan's original survey was transposed to "A lot of problems in the industry today come from the decline in traditional contract structures, where the government regulates and the industry follows" in the survey of this research as shown in Table 40.1. As both questions measure the construct of hierarchy, the subject's worldview would be more hierarchical than egalitarian, if he/she agrees to the questions.

Table 40.1 Example of survey items measuring individuals' cultural worldviews

Index	Kahan 2012	NASC survey 2016
HTRADSTR	A lot of problems in our society today come from the decline in the traditional family, where the man works and the woman stays home.	A lot of problems in the industry today come from the decline in traditional contract structures, where the government regulates and the industry follows.
ERADEQ	We need to dramatically reduce inequalities between the rich and the poor, whites and people of color, and men and women.	We need to dramatically reduce inequalities between big and small companies, government agencies and industries, and OEMs and manufacturers.
ESTDDIS		Discussions regarding standards should be open to anybody in the industry.
IINTRFER	The government interferes far too much in our everyday lives.	When it comes to standards, the government interferes far too much in companies' decisions.
SLIMCHOI	Government should put limits on the choices individuals can make so they don't get in the way of what's good for society.	The government should put limits on the choices companies can make so they don't get in the way of what's good for the industry.
IINNOSPD		The government's regulations slow down technology innovation for the industry.

The survey items for this research (right) was adapted from Kahan's survey items [6] (left). The index indicates whether the survey item measures hierarchical (H) or egalitarian (E), and individualistic (I) or solidaristic (S). For example, if a subject agrees to the item with an "H" in its index, it means that the subject possesses more hierarchical than egalitarian cultural worldview. A blank cell means that the survey item was developed for this study without the adaptation of Kahan's item

Responses were recorded using a 7-point Likert scale for 29 survey items, ranging from strongly agree (1) to strongly disagree (7). We also measure demographic information including gender, age, years of professional experience, language, ethnicity, religion, education level, and job position. Finally, we collected information on member affiliations, size of the organizations that the committee members belong to, and job functions with their organizations because it was anticipated that these three variables would be associated with the cultural worldviews of each individual.

40.3.3 Statistical Analysis

We examined whether our survey items reliably measured cultural worldviews using Cronbach's α, an indicator of whether these survey items measure the same underlying construct. We also conducted a correlation analysis to examine relationships among survey items, identify what are the core elements among survey items, and verify what they actually measure.

40.4 Results

40.4.1 Survey Results

Data were collected for 55 members (26% response rate). A plurality (49%) of respondents belonged to part manufacturers, and almost half (49%) of respondents belonged to organizations that had more than 500 employees. Thirty-one percent of the respondents were responsible for upper management roles in their organizations, and 22% and 25% of respondents described themselves as middle management and trained professional, respectively. More than half (61%) of respondents claimed that they had been involved in the industries for 30–50 years (Table 40.2).

Subjects' scores along each dimension, calculated from the responses to 29 survey questions, are plotted in two-dimensional space delineated by group and grid axes (Figs. 40.3 and 40.4). The majority of subjects (75%) were individualistic. Twenty-two subjects demonstrated hierarchical-individualistic worldviews, and 17 subjects fell into egalitarian-individualistic quadrant. Four members scored 0 in hierarchy-egalitarianism measure, and thus fell exactly on the group axis, which means their worldviews were neither hierarchical nor egalitarian.

Although sample size was too small to draw statistically valid inferences, the underlying trends are instructive. Regarding the organizational types, we observed the following tendencies (Fig. 40.3): the average cultural worldview of experts working for part manufacturers was egalitarian-individualistic; the average OEM manufacturer also showed an egalitarian-individualistic worldview, but higher group tendency than that of part manufacturers; the average respondent from government agencies was located in the hierarchical-communitarian quadrant; and the average part distributor had a hierarchical-individualistic worldview.

As for the organizational sizes (Fig. 40.4), the average respondent from bigger organizations showed higher group tendency, meaning that their worldviews were more communitarian. Also, this group of people shows higher grid tendency (more hierarchical), on average, than the people from the organizations with the smaller sizes.

We also examined the relationship between subjects' years of experience and their cultural worldviews, although no consistent trends were observed from our results.

Table 40.2 Statistics for the sample. $N = 55$

Affiliation	*n* (%)	Size	*n* (%)	Function	*n* (%)	Years	*n* (%)[a]
Part manufacturers	27 (49)	>500	27 (49)	Upper management	17 (31)	<10	2 (4)
Inspection lab	0 (0)	100–500	7 (13)	Middle management	12 (22)	10–19	7 (13)
Distributor	5 (9)	20–99	17 (31)	Staff	4 (8)	20–29	10 (19)
OEM	12 (22)	10–19	2 (4)	Trained professional	14 (25)	30–39	20 (37)
Government agency	5 (9)	<10	2 (4)	Consultant	5 (9)	40–50	13 (24)
Other	6 (11)			Other	3 (5)	>50	2 (4)

[a]One subject did not respond as to years of experience

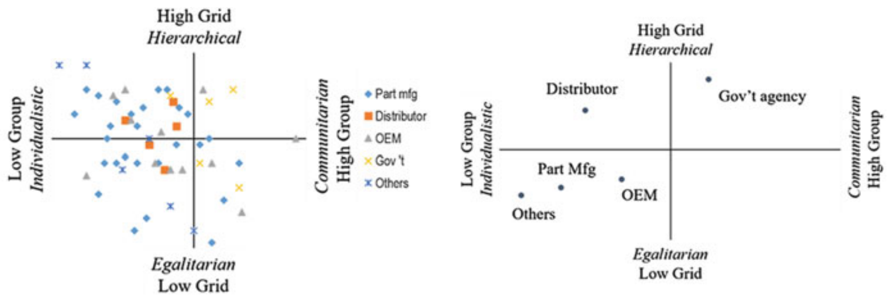


Fig. 40.3 Cultural cognition map based on organizational types that the subjects belong to. Mapping of individuals' scores from the survey measuring worldview with their affiliations (*left*) vs. average scores per affiliation category (*right*)

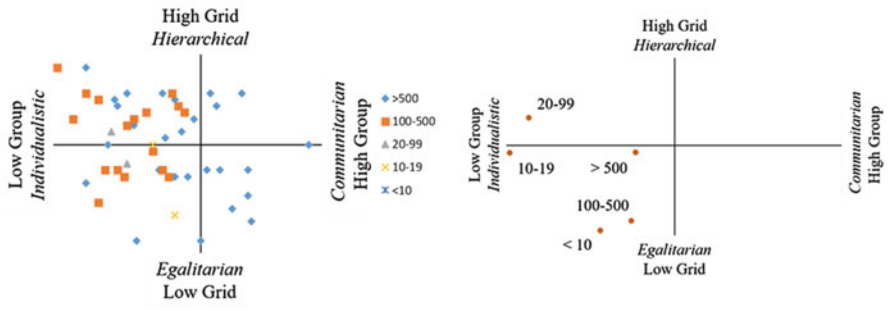


Fig. 40.4 Cultural cognition map based on sizes of organizations that the subjects belong to. Mapping of individuals' scores from the survey measuring worldviews with organization size (*left*) vs. average scores per size category (*right*)

40.4.2 Statistical Analysis

For the survey items measuring communitarian-individualistic worldviews, Cronbach's α was 0.89, indicating their strong internal consistency. In contrast, Cronbach's α for the hierarchy axis was found to be low: 0.45, indicating weak internal consistency among the survey items measuring hierarchy-egalitarianism; however, sample size was small and variance along this dimension may have been limited by response bias (more respondents from larger organizations).

40.4.3 Pearson Correlation

We examined bivariate correlations among survey responses to better understand the nature of our data. We identified two clusters of items in which bivariate correlations between all items were greater than 0.60 (Table 40.3). Cronbach's α was 0.92 for the survey items in the individualism scale of cluster 1, which indicates even stronger internal consistency than all the survey items measuring individualism-communitarianism (0.89). It appears that those items have the word, "government" in common.

40.5 Discussion

40.5.1 Summary of Preliminary Findings

As an initial step of this research, we first hypothesized that the cultural worldviews of standards committee members could be categorized into four quadrants on a 2D space adapted from Kahan's cultural cognition hypothesis. We found that most respondents answered questions consistent with an egalitarian-individualistic cultural worldview, characterized by Douglas as "market," or as "the culture of the entrepreneurial professionals, they tend to perceive the situation as risk when free trade or their efforts for innovation is restricted and inhibited" [8]. People from part manufacturers and OEMs belong to this category based upon the average scores of the survey (Fig. 40.3).

People from the government agencies are located in the first quadrant, consistent with their role as a hierarchical-communitarian group. Douglas and Wildavsky [1] associated this quadrant with a decision-making process that relies on the method of "successive limited comparison," in which each potential outcome from the decision-making process is compared to the status quo [16] – that is, "bureaucratic decision-making."

Table 40.3 Survey items constructing clusters with strong correlations

	Index	Survey items
Cluster 1	IFIX	When it comes to standards, we'd all be a lot better off if the government spent less time trying to fix everyone's problems.
	IGOVWAST	When it comes to standards, government regulations are almost always a waste of everyone's time and money.
	IINTRFER	When it comes to standards, the government interferes far too much in companies' decisions.
	IMKT	When it comes to standards, free market – Not government bodies – Are the best way to reach agreement on better system.
	SPROTECT	The government should do more to advance society's goals, even if that means limiting the freedom and choices of individual companies in the industry.
Cluster 2	IPRIVACY	The government should stop telling companies how to run their business.
	IPROTECT	It's NOT the government's business to try to protect the industry from bad decisions.
	IRESPON	The industry works best when it lets companies take responsibilities for their own actions without telling them what to do.

Finally, we found that the average part distributor was located in the hierarchical individualist quadrant, consistent with their roles as an intermediary between two different hierarchical levels that must nevertheless seek profit.

40.5.2 What the Survey Measures

We adapted this survey from Kahan's survey questions; nevertheless, we must verify that the adaptation is valid. Did the adapted survey measure what it was designed and intended to measure to assign the individual's cultural way of life on the two-dimensional space defined by the group and grid dimensions? Examining Cronbach's α is one way of answering this question, as it is an indicator of the consistency of survey items on whether or not they measure the same construct. Our results suggest that the items to measure individualistic versus communitarian worldviews are consistent.

Despite a high value of consistency, items indexed by this axis largely included discussion of government intervention, consistent with van der Linden's critique [15] that Kahan's items may simply index trust in government. Indeed, two clusters composed of strongly correlated items were found among the items measuring individualism. The first cluster, which consists of five survey questions, asks the subjects' opinions on whether or not the government's involvement in standardization is appropriate for the industry's well-being. Cluster 2, consisting of three questions, asks if it is appropriate for each individual company to make business decisions on their own, independent of an authority's involvement. Furthermore, items in these clusters are correlated (the weakest correlation between items in these clusters is $r = 0.631$, p-value < 0.001).

In contrast, subjects did not respond as consistently to items designed to measure hierarchy versus egalitarianism. This may be due to small sample size, skewed data, or the absence of a strong effect of hierarchy. Furthermore, there were no significant bivariate correlations between the survey items designed to measure hierarchy or egalitarianism.

40.5.3 Directions for Future Work

Future work will further refine the survey instrument as follows: First, more work needs to be performed to verify, assess, and refine survey items, to address the concerns related to their validity. This includes targeted cognitive interviews of subjects to get a better understanding of their interpretations of survey items. Second, outcome measures that we can predict, such as risk perception and blame attribution, preference regarding the design of standards, or use of standards, will be developed for inclusion in the survey. This will help us to understand what factors influence the decisions that professional groups have to make.

40.6 Conclusion

In this paper, we introduced and discussed preliminary work for a study on cultural worldviews on technical expert committee. Our findings provide promising preliminary evidence suggesting that experts' cultural worldviews may be associated with their backgrounds, including the type of company and sizes of the organizations to which they belong. Future work will focus on refining the survey and collecting more data to validate this approach, as well as identifying proper outcome measures to understand the decision-making mechanisms of the technical experts.

References

1. Douglas M, Wildavsky A (1982) Risk and culture: an essay on the selection of technological and environmental dangers. University of California Press, Berkeley/Los Angeles
2. Douglas M (1986) How institutions think. Syracuse University Press, Syracuse
3. Broniatowski DA (2015) Does systems architecture drive risk perception?" In: Cetinkaya S, Ryan JK (eds) Proceedings of the 2015 industrial and systems engineering research conference, Institute of Industrial Engineers, Nashville.
4. Moses J (2002) The anatomy of large scale systems. March 25, In: Proceedings of the MIT ESD Symposium, Cambridge, MA

5. Broniatowski DA, Moses J (2014) Flexibility, complexity, and controllability in large scale systems. In: CESUN 4th International Engineering Systems Symposium, Hoboken, June 2014
6. Kahan DM (2012) Cultural cognition as a conception of the cultural theory of risk. In: Roeser S, Hillerbrand R, Sandin P, Peterson M (eds) Handbook of risk theory. Springer, Dordrecht, pp 725–759
7. Douglas M (1970) Natural symbols: explorations in cosmology. Barrie & Rockliff, Cresset Press, London
8. Douglas M (1994) Risk and blame: essays in cultural theory. Routledge, New York
9. Rayner S (1992) Cultural theory and risk analysis. In: Krimsky S, Golding Eds D (eds) Social theories of risk. Praeger, New York, pp 83–115
10. Thompson M, Ellis R, Wildavsky A (1990) Cultural theory. Westview Press, Boulder
11. Kahan DM, Jenkins-Smith HC, Braman D (2011) Cultural cognition of scientific consensus. J Ris Res 14:147–174. On-line publication http://dx.doi.org/10.1080/13669877.2010.511246
12. Dake K (1991) Orienting dispositions in the perception of risk: an analysis of contemporary worldviews and cultural biases. J Cross Cult Psychol 22:61
13. Peters E, Slovic P (1996) The role of affect and worldviews as orienting dispositions in the perception and acceptance of nuclear power. J Appl Soc Psychol 26(16):1427–1453
14. Slovic P (2000) The perception of risk. Earthscan Publications, London/Sterling
15. Van der Linden S (2016) A conceptual critique of the cultural cognition thesis. Sci Commun 38 (1):128–138
16. Lindblom CE (1959) The science of “muddling through”. Public Adm Rev 19(2):79–88

Chapter 41
The Flexibility of Generic Architectures: Lessons from the Human Nervous System

David A. Broniatowski and Joel Moses

Abstract Many engineered systems are biologically inspired. In this paper, we examine the structure of the human nervous system with an eye toward understanding how its internal architecture may inform the design of large-scale engineered systems. Specifically, we examine four types of "generic" architectures – tree-structured hierarchies, layered hierarchies, diffuse networks, and teams. We observe all four types of these hierarchies in the human nervous system. Consistent with prior theory, tree-structured hierarchies are relatively inflexible, but simple and easy to control. Layered hierarchies are moderately flexible, more complex, yet still largely controllable. Diffuse networks are easy to describe and therefore relatively simple yet flexible; however, they can lead to unexpected emergent behaviors undermining controllability. Finally, team structures are extremely flexible, but can lead to instabilities. Implications for system design are discussed.

Keywords Layered hierarchy • Tree • Diffuse network • Grid • Team • Architecture

41.1 Introduction

The human nervous system, and especially the human brain, is among the most complex systems known to exist. Despite its complexity, the nervous system is also extremely flexible, enabling the human organism a range of behaviors across a wide variety of environments, cultures, and other contexts. Some have argued that the complexity of the human brain is beyond human comprehension (e.g. biologist Jack Cohen and mathematician Ian Stewart famously declared "If our brains were simple enough for us to understand them, we'd be so simple that we couldn't." [1]). Nevertheless, we believe that the structure of the human nervous system contains

D.A. Broniatowski (✉)
The George Washington University, Washington, DC, USA
e-mail: broniatowski@gwu.edu

J. Moses
Massachusetts Institute of Technology, Cambridge, MA, USA
e-mail: moses@mit.edu

© Springer International Publishing AG 2018
A.M. Madni et al. (eds.), *Disciplinary Convergence in Systems Engineering Research*, https://doi.org/10.1007/978-3-319-62217-0_41

lessons regarding the relationship between a system's flexibility and its architecture. Although a detailed, mechanistic understanding of the mind and its biological substrates remains at the frontiers of scientific research, our key goal is to understand how the structure or organization of systems can help designers to create, modify, and operate complex systems in the most effective manner. To that end, we look at the structure of the human nervous system for inspiration.

In this paper, we examine how four types of system architectures that are commonly found in complex engineered systems manifest in the human nervous system. Elsewhere [2], we have explored the properties of these four "generic" architectures, and especially their flexibility (how easily a change is made to the system), descriptive complexity (the ease with which the system is described), and rework potential (i.e. the extent to which decisions made by actors within the system can be revisited or countermanded by other actors – in effect, rework potential captures the need for consensus or compromise when making decisions). Despite the fact that engineered systems are designed by humans and the nervous system is not, we claim that comparable architectures across these widely disparate domains manifest comparable lifecycle properties.

41.1.1 Why Study "Generic" Architectures?

The importance of systems architecture has increased due to engineering developments in the past 50 years. In a number of engineering fields, such as electronic engineering and biological engineering, the research and development of new products and processes emphasized smaller and smaller components, such as transistors in a chip and cells in the body. At the same time, the total number of components in engineered systems grew substantially so that some engineered systems, such as the Internet, had millions and even billions of components. Software systems frequently have millions of lines of code. Furthermore, software and hardware are now virtually indistinguishable with the advent of the "Internet of Things" (i.e. embedded software).

Such engineered systems are inherently complex. One source of this complexity is when the connections between the parts are architecturally "messy" (i.e. difficult to describe) [2], such that it is nearly impossible to determine the system's output for a given input. Thus, trying to achieve a desired behavior in large-scale complex systems is difficult. Trying to do so when the system's goals are changing, as they usually are in large-scale systems, is even more difficult. We do not claim to explain here the full behavior of large-scale or even medium-scale systems. Nor do we claim to understand the advantages and disadvantages of the architecture of "real" systems. Indeed, "real" systems change sufficiently often that any attempt to describe their architectures precisely will not be accurate for long. Our emphasis is therefore on understanding "generic," or idealized, system architectures. We believe that a deep understanding of ideal or generic architectures, and the advantages and disadvantages of such architectures, will help, in part, in understanding how a real system behaves and how one can change it to achieve desired changes in its behavior.

41.2 Four Generic Architectures

We use the term "system architecture" to refer to overall structure or organization of a system. A "generic architecture" denotes a system architecture whose pattern of interconnections is similar throughout the system. For small-scale systems we shall discuss team structures, where every node is usually connected to every other node; however, team structures are not normally the overall structure in large-scale systems. For large-scale systems, we emphasize three types of generic architectures – tree structures, layered structures, and undifferentiated networks. Importantly, several generic architectures may be present in the same system, only at different levels of abstraction. For example, we previously discussed how all four generic architectures were used by the US Postal Service [2]. Although we do not claim that these four architectures are mutually exclusive or collectively exhaustive, they highlight important trade-offs between flexibility and descriptive complexity (in [2], we analyze these trade-offs mathematically). We claim that these architectures possess these trade-offs across multiple domains.

41.2.1 Why Study "Generic" Architectures?

Systems can be modeled using a graph-theoretic representation composed of nodes and edges. Here, nodes represent information processors (i.e. individual, but interconnected, units that send informational signals), whereas edges represent the flow of information between these units. Elsewhere, [2] we argued that the pattern of these interconnections can have great impact on one's ability to modify the system during its useful lifetime. Furthermore, these interconnection patterns are very important when the number of nodes is large such as in the following examples:

(a) A firm with over one hundred thousand employees
(b) A jet airplane with millions of parts
(c) A software system with over ten million lines of code
(d) The human nervous system with tens of billions of neurons and hundreds of trillions of interconnections between them

Note that examples a, b, and c are systems and organizations that are designed by humans. Example d, the human brain, is created through a complex biological process involving genes as well as environmentally induced changes, such as the changes that occur when one learns grammar rules, new vocabulary in a language, or memories of a human face. It is a fundamental premise of our work that architectural properties of complex systems generalize across these widely diverse contexts. Whereas prior work [2] has focused on systems designed by humans, in this paper, we will focus on the structure of the human nervous system.

41.2.2 Tree-Structured Hierarchies

Nodes in a pure tree structure, other than the top node, have exactly one parent node. Pure tree structures are hierarchies that have vertical connections only and no horizontal ones. Impure tree structures will have additional interconnections that lead to nodes having more than one parent node. Such additional interconnections will usually be relatively ad hoc. An organization with very many ad hoc interconnections will tend to be very complex and hard to modify. (Note that tree structures are not, in general, hard to modify; however, too many changes can increase the descriptive complexity of the structure so much that the consequences of a modification may be prohibitive to track [2], Fig. 41.1).

Other examples of tree structures include:

(a) Parse tree for a sentence in English
(b) Decomposition of the design of a physical system, such as a car, into modules
(c) Structure of branches of a physical tree

41.2.3 Layered Hierarchies

Unlike tree-structured hierarchies, nodes in a generic layered hierarchy are usually connected to one or more nodes in the layer immediately below them as well as nodes in the layer immediately above them. Nodes can also connect with nodes at the same layer, which is not the case in pure tree structured systems. However, layered structures are still hierarchies. Nodes at the same layer can form teams. These teams are indicated via horizontal interconnections. Layers are usually related to abstractions, and new layers can be formed by creating new abstractions. For example, a medieval trade guild is usually organized as a layered hierarchy with apprentices working on low-level tasks, masters working on high-level designs, and journeymen working on translating one to the other (Fig. 41.2).

Examples of layered structures include:

(a) The architecture of the Internet
(b) Automobile platforms as a layer
(c) Integers–rationals–polynomials in x with rational coefficients (each of the three layers can be infinitely large)
(d) The hierarchy of programming languages in a software system (the number of programs in each layer is potentially infinite)

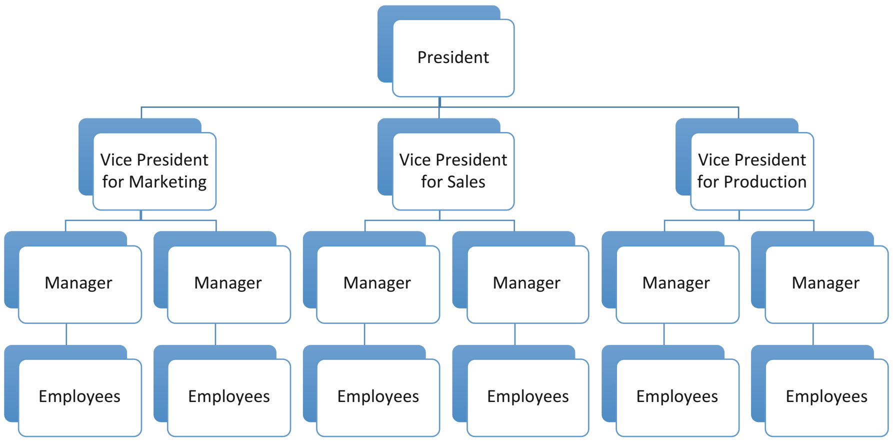


Fig. 41.1 An example of a tree-structured human organization

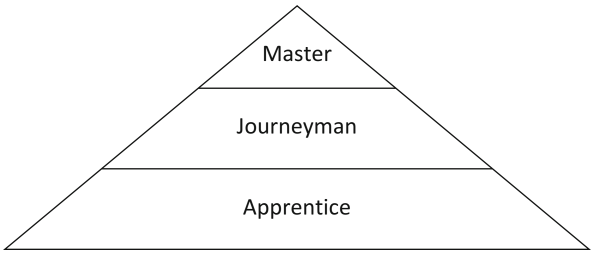


Fig. 41.2 An example of a layered human organization

41.2.4 Undifferentiated (e.g. Grid) Networks

Nodes in an undifferentiated network, such as a grid, connect to all or most proximal nodes. Unlike tree and layered structures, these networks are not hierarchical. For example, generic grid networks are flat. In such networks, each node responds to its immediate environment and sends information to its neighbors, potentially leading to emergent behaviors. For example, traffic jams may form on a grid where nodes represent the intersection of streets, and edges represent the flow of traffic between these intersections. Figure 41.3 shows an example of a traffic grid defined on a nearest-neighbor network.

In general, such networks need not be strict grids, and the length of the interconnections will vary, sometimes greatly. Examples of such distributed networks include:

(a) The spread of disease, as modeled by cellular automata [4]
(b) Models of the weather
(c) Percolation of liquid through a porous material [5]

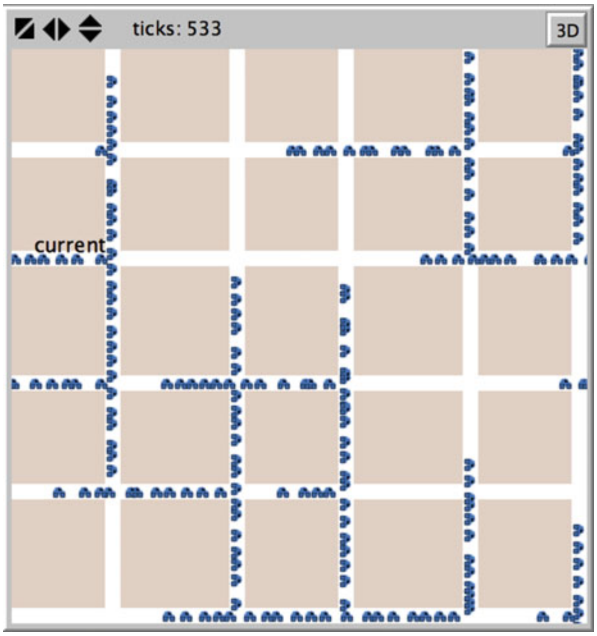


Fig. 41.3 A traffic grid network generated using the NetLogo software [3]

41.2.5 Teams

In addition to the three generic architectures mentioned above that can be used in large-scale systems, we also include team structures, largely used in human organizations. Members of a team are connected to every other member, yet they lack an explicit hierarchy (other than a possible team leader, who seeks agreement rather than control). This can make reaching consensus difficult. One of the advantages of a team structure is that it can often cope with high rates of change. For example, when one team member is ill, the others can usually pick up the slack quickly, albeit at some loss in performance. A major disadvantage of the team structure is that people cannot have close interactions with a large number of others. This greatly limits the number of members in a team of humans (Fig. 41.4).

41.3 Generic Architectures in the Human Nervous System

The fundamental unit of the human nervous system is the neuron – a cell that processes information and uses electrical and/or chemical means to send it to other cells. Specifically, the output of a neuron is an electrical signal that is carried along an axon to other neurons. Neurons are connected to one another in several relationships, including hierarchically and laterally. Furthermore, the human brain is composed of roughly one hundred billion neurons and on the order of a quadrillion

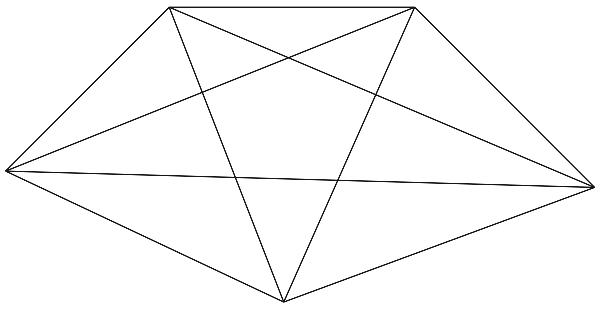


Fig. 41.4 A team structure with five nodes and interconnections between each pair of nodes

interconnections among them. Finally, the brain is just one part of a larger system that extends throughout the body, consisting of the spinal cord and other nerves. In this section, we describe how this huge system uses all four generic architectures for large-scale systems – trees, layers, diffuse networks, and teams.

41.3.1 Tree Structures in the Human Nervous System

Swanson [6] describes the hierarchical organization of the somatic (i.e. voluntary) motor system as a tree-structured hierarchy in which higher levels actuate muscle groups in the lower levels in stereotyped or patterned ways. Here, a group of muscle cells might be activated by a "motor neuron" – a neuron whose axon terminates in this group. These motor neurons are not activated individually; rather, several motor neurons form a "motor neuron pool" which, when activated, extends or flexes an entire muscle. These motor pools are, in turn, controlled by "central pattern generators" – groups of motor neuron pools which ensure that when a muscle on one side of a limb flexes, the corresponding muscle on the other side of the limb extends. These pattern generators are further controlled by "locomotor pattern initiators," which control groups of muscles, all of which flex or extend in unison. Finally, at the top of the hierarchy are "locomotor pattern controllers" that may be activated in response to specific actions, as shown in Fig. 41.5.

Like other tree-structured hierarchies, the motor system is highly controllable yet relatively inflexible. Such controllability is necessary to ensure that desired behaviors are carried out in a repeatable manner. On the other hand, if a given part of the system is destroyed or disrupted, such as when the spinal cord is severed, the system may not fully recover, leading to paralysis of the individual below the point where the cord is cut and, frequently, the atrophy of the associated muscles.

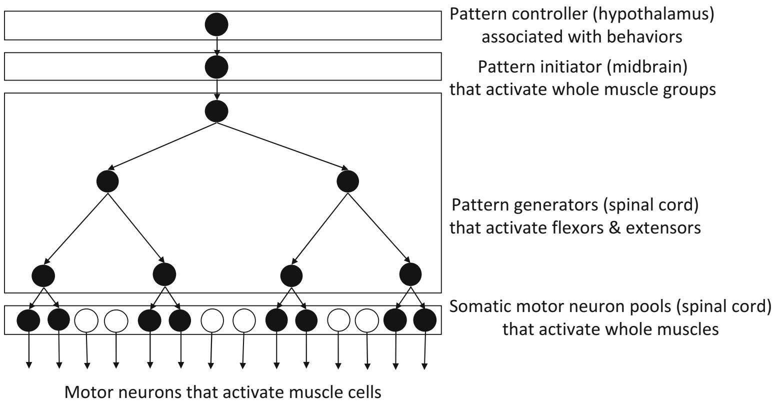

Pattern controller (hypothalamus) associated with behaviors
Pattern initiator (midbrain) that activate whole muscle groups
Pattern generators (spinal cord) that activate flexors & extensors
Somatic motor neuron pools (spinal cord) that activate whole muscles
Motor neurons that activate muscle cells

Fig. 41.5 Structure of the somatic motor system (Adapted from [6])

41.3.2 Diffuse Networks in the Human Nervous System

The neurons of cnidarians, such as jellyfish and sea anemones, are distributed roughly uniformly throughout most of their bodies, in an arrangement known as a “nerve net.” Nerve nets are nonspecialized “nearest neighbor” arrangements that diffuse neural signals (such as those initiated by physical contact with a food particle) throughout the body equally in all directions. Furthermore, the strength of the stimulus is inversely proportional to the distance from the stimulus. In humans, such nerve nets are found in limited, yet important parts of the brain. Specifically, Swanson [6] describes the “amacrine” cell layer of the retina (amacrine cells do not have axons but instead have lateral bidirectional connections to other nearby cells), the granule cell layer of the olfactory bulb, and the lining of the human digestive system as nerve nets.

One defining feature of nerve nets is that they are made of amacrine cells, which may send signals laterally and therefore have no hierarchy. Thus, like grid networks, nerve nets are distinguished by their lack of cephalization, or top-down control. As the neurons in a nerve net are relatively undifferentiated, their behavior is subject almost entirely to responses to external stimuli. These are among the least complex nervous system structures and their behaviors are consequently less controlled. On the other hand, they are quite flexible – injured cnidarians are often able to regrow lost limbs or reposition existing limbs to maintain radial symmetry.

41.3.3 Team Structures in the Human Nervous System

Brodmann [7] analyzed the large-scale regions of the cerebral cortex a century ago. These regions relate, for example, to the input senses, such as vision, and outputs, such as speech, as well as task planning. When the organism encounters an event in the external world, such as when a large animal passes by causing a loud noise, several stimuli activate these regions, and these stimuli must be integrated to make sense of the event. These regions are interconnected with one another and must be fused into a common "consensus" representation. Thus, when performing sensory integration, the brain uses a form of team structure (e.g. [8], in which multiple sensory modalities must "agree" on the meaning of a stimulus).

One major computational challenge faced by cognitive scientists and researchers in artificial intelligence has been trying to determine how these different modes achieve agreement in the absence of hierarchical structure. One leading modeling paradigm represents each mode as a "daimon" favoring a certain outcome, with consensus emerging from pairwise negotiations between preferred interpretations [9]. In some cases, these negotiations may not converge, leading to multistable percepts – that is, disagreement between sensory modalities and the inability to form a single stable interpretation [10]. Similarly, the sensory system can be "fooled" by conflicting information between modes, such as when a passenger on a train receives visual information indicating motion but no corresponding stimulus from the vestibular system indicating acceleration. The system may conclude that the subject is moving when, in fact, another train is moving relative to the subject but the subject remains stationary – a phenomenon known as "vection." The illusion resulting from this conflict frequently causes motion sickness.

Like other team systems, the sensory integration system is quite flexible – humans are able to infer several robust conclusions from limited sense data. In addition, the brain has been shown to exhibit significant neuroplasticity such that, if one sensory modality is lost, others can, to some degree, compensate (e.g. individuals with limited vision frequently have more sensitive hearing). Unlike hierarchical systems, the sensory system has limited controllability as manifested by multistable perception. A "decision" made by a single sensory modality is subject to revision by other modes and, if they disagree, the organism may not be able to converge on a single interpretation.

41.3.4 Layered Hierarchies in the Human Nervous System

For us, the interesting aspect of brain architecture is in the intermediate structure of interconnections. If one flattens the highly convoluted cerebral cortex, one obtains a sheet that is a few millimeters deep. The depth of the cerebral cortex is composed of six horizontal layers of neurons. Some questions arise about the layered architecture of the cerebral cortex. For example, what is its purpose? Like many questions about the human brain, we do not currently know the answer, but Jeffrey Hawkins presents an interesting theory [11]. He notes that the cortical neurons are located in vertical columns that effectively cross the six layers. His theory is that neurons at a layer higher than a given neuron, but in the same column, are operating at a higher level of abstraction. For example, one does not immediately see a chair with one's

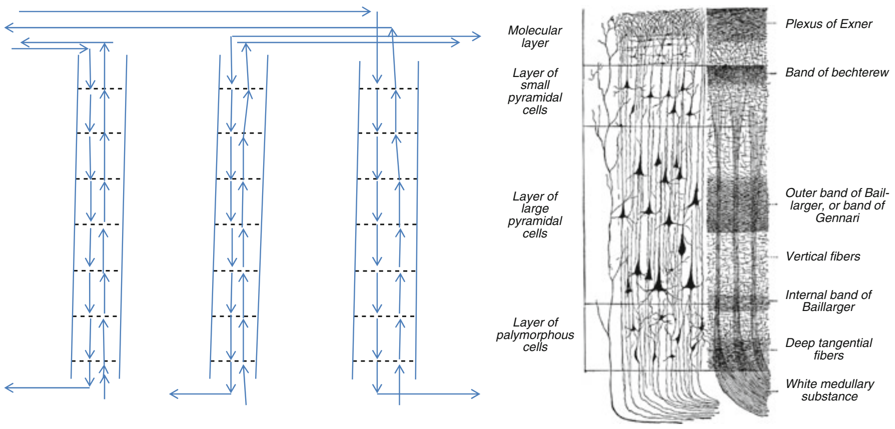

Molecular layer
Layer of small pyramidal cells
Layer of large pyramidal cells
Layer of palymorphous cells
Plexus of Exner
Band of bechterew
Outer band of Bail-larger, or band of Gennari
Vertical fibers
Internal band of Baillarger
Deep tangential fibers
White medullary substance

Fig. 41.6 *Left*: Schematic representation of columnar structure of the neocortex. *Right*: Layers in the human cortex. Vertical fibers that pass information both up and down the hierarchy, whereas pyramidal cells also possess horizontal connections (By Henry Vandyke Carter – Henry Gray (1918) Anatomy of the Human Body, Bartleby.com: Gray's Anatomy, Plate 754, Public Domain, https://commons.wikimedia.org/w/index.php?curid=541599)

eye. Instead, optical inputs to the eye go through many phases of analysis in the brain before one has something like an integrated conception of a chair. A key to Hawkins's theory is that going up a layer yields predictions about future inputs to that layer. If the predictions hold up, then one sends an electrical pulse to neurons in the layer above and continues processing the inputs. If, however, the prediction does not hold, then downward connections are activated so that one can analyze new inputs to yield abstractions and predictions, which may hold up in the future (Fig. 41.6).

Connections that go upward or downward usually do so in a column of neurons [12]. Some connections will also go horizontally from one column to another within a given region. Connections that move to another region in the brain do so in a bundle of axons. These bundles carry signals over relatively long distances, both within the brain and from the brain to other parts of the body. These bundles are encased in a sheath composed of myelin. The myelin sheath permits the electrical signals in the axon bundles to move quickly over long distances.

Although there are six layers, only three of them are "processing layers" [13], whereas the remaining three are input or output layers (elsewhere [14], we have emphasized the "magical properties" of three layers – many engineered systems also have three layers). As one likely needs a layer for inputs from other regions of the brain, and a layer for outputs, one probably does not get a full six layers of processing at any column in the cortex. Furthermore, we know that different regions (e.g. the vision regions V1 and V2) of the cortex work on visual information at increasingly higher levels of abstraction (the "top" of this hierarchy is one node in the team network discussed in Sect. 41.3.3). Hawkins further emphasizes the role of patterns: neurons in higher layers fire in response to patterned signals from inputs received from lower level layers Thus, there is an advantage for regularity of structure, and different regions of the brain can work at different levels of abstraction.

Other animals often have layered cortices. For example, Hawkins notes that dolphins have a large brain, but only three layers of neurons. Likely they can remember many parts of the oceans they traverse, but their language capabilities are nowhere near ours. Swanson [6] gives additional examples of organisms that have just one neuron or just one or two layers of neurons, and indicates the limitations of such architectures.

Cortical layering enables flexibility. For example, Ballard [13] treats lower level neural signals in the cortex as state variables and primitives that serve as inputs to higher layers in a manner analogous to how computer programs use abstraction to enable a virtually infinite number of behaviors. In general, the hierarchical nature of the cortex allows for abstracting across a range of stimuli. Thus, the cortex allows humans to anticipate and predict outcomes based on incomplete sensory data, contributing to fast decision-making. In addition, the cortical structure tends to be relatively flexible – for example, significant bodies of literature on “analogical transfer” (e.g. [15]) and “cognitive flexibility” [16, 17] indicate that prior abstractions can be adapted to new situations to enable the human organism to cope with changing environments. These adaptations come at the cost of significant complexity – although the generic structure of the neocortex is increasingly well-understood, replicating its behavior remains beyond our capabilities.

41.4 Discussion

The above discussion shows how different components of the human nervous system use different architectures. Taken as a whole, the nervous system uses all four of the generic architectures for large-scale systems that we have been discussing. Beyond these implications for system design, the benefits and drawbacks of each of these structures can be assessed by examining the ecological niches for organisms that display one architecture to the relative exclusion of others. For example, Tinbergen [18] describes the instinctive behavior of the male three-spined stickleback fish as tree-structured (Fig. 41.7).

Such organisms exist in an environment that requires relatively little behavioral adaptation. In contrast, hydras possess a relatively limited repertoire of behaviors that nevertheless allow for significant adaptation to stimuli of several different shapes and sizes. Many different species use of several different sensory modalities, demonstrating the flexibility of a scheme that has been widely used across the animal kingdom. Finally, humans are among the few organisms to make use of extensive layered hierarchical structure in the neocortex; however, this is an adaptation that has created a significant evolutionary advantage, leading to the role of the human being as the dominant organism on Earth. Importantly, the human nervous system architecture uses all of these in a hybrid mode. The design

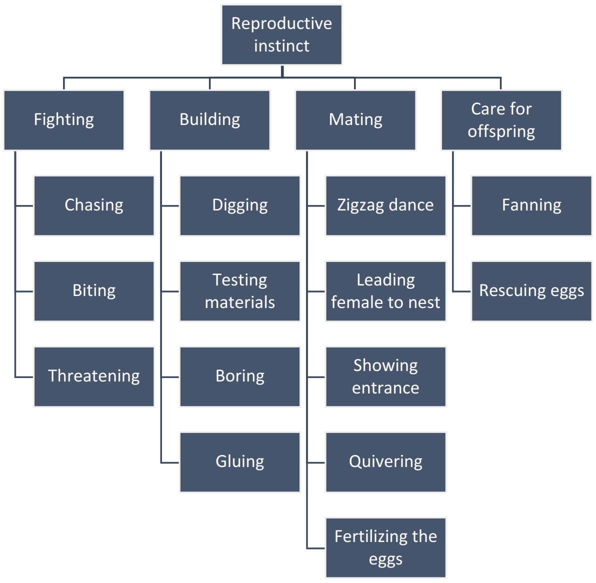


Fig. 41.7 Stereotyped behaviors of the three-spined stickleback fish, described by a tree-structured hierarchy

of complex systems has much to learn from how to combine these different architectures in a manner that is most conducive to a system's needs and environment.

41.4.1 Implications for the Design of Complex Engineered Systems

The primary contribution of this paper is the deduction of engineering design principles from the architectures found in the human nervous system. Consistent with our prior theory [2], the analysis presented above indicates the strengths and weaknesses of generic system architectures. Specifically, tree-structured hierarchies are straightforward to control yet relatively inflexible. A case in point is lower body paralysis to the legs that might result from a disruption to the spinal cord. Although it is frequently possible for the human body to adapt to such disruptions to some degree (e.g. by using assistive devices), these ad hoc changes require extra complexity (as in the incorporation of new equipment). Undifferentiated networks are much more flexible due to their redundancy (and consequent low complexity). For example, a hydra that has lost a limb can still function and, in some cases, grow the limb back with minimal if any disruption to its ability to function. This flexibility comes at the cost of centralized control. Teams are also extremely flexible. For example, a patient who loses his or her sight can often compensate with other senses. Like undifferentiated networks, teams lack a centralized controller, limiting their size. Finally, layered hierarchies strike a balance between these different extremes, providing more flexibility, and less control, than tree structures but less flexibility and more control than undifferentiated networks and teams.

41.4.2 Conclusions

The world is increasingly reliant on large-scale sociotechnical systems. Examples of such systems include health care systems, manufacturing systems, energy systems, and the environment. All such systems are dependent on interactions with society. Public policy plays a key role in sociotechnical systems. While the technologies underlying different sociotechnical systems vary, there are fundamental issues that such systems have in common. These include the various "ilities" [19] such as flexibility, robustness, resilience, safety, and sustainability. Structure of sociotechnical systems is also a key issue, as is complexity. Thus, although the various large-scale sociotechnical systems are different in behavior, they have much in common.

The four generic architectures considered in this study are team structures, tree structures, layered structures, and diffuse networks. There are advantages and disadvantages to each such architecture. Human teams are limited in size, but they can often accommodate many small internal and external changes. Tree structures can handle a great variety of systems, but it may become overly complex after making numerous small changes, and can then be difficult to modify further. Layered structures are not generally applicable, but when they can be used they will likely accommodate many changes very well. There are various classes of networks. The diffuse networks we consider here are quite general and can handle simple automatic behaviors with significant flexibility, although with some potential for emergence – these networks may become difficult to control after some changes in the internal structure.

The human nervous system uses several of these generic architectures. The motor system is arranged as a tree structure. Parts of the retina, olfactory bulb, and viscera are organized as diffuse networks. Several sensory modes must work together to achieve consensus on an interpretation of a set of stimuli. These modes are essentially all connected to one another. Relatively long connections between regions are carried in bundles that are surrounded by myelin-based wrapping to speed up the transmission of signals. The cortex, when unwound, is composed as a sheet largely made up of six layers of neurons with a thickness of a few millimeters. We do not currently know what role the layers play, but one theory is that neurons at one layer recognize signals at a higher level of abstraction than the level of abstraction recognized by neurons at a lower layer that are connected to it.

We do not claim that the four architectures discussed here are neither mutually exclusive nor collectively exhaustive; indeed, we find that they are all present, to some extent, within one organism. Furthermore, we have noted that they tend to appear frequently in multiple contexts. The fact that they have been heavily used by animal biology – certainly a flexible system – indicates that we may be able to gain inspiration from these sources. Elucidating the strengths and weaknesses of the different architectural approaches can help us to achieve this design goal.

References

1. Cohen J, Stewart I (2000) The collapse of chaos: discovering simplicity in a complex world. Penguin, London
2. Broniatowski DA, Moses J (2016) Measuring flexibility, descriptive complexity, and rework potential in generic system architectures. Syst Eng 19(3):207–221
3. Tisue S, Wilensky U (2004) Netlogo: a simple environment for modeling complexity. In: International conference on complex systems [Internet]. Boston, pp 16–21
4. Wolfram S et al (1986) Theory and applications of cellular automata, vol 1. World scientific, Singapore
5. Aharony A, Stauffer D (2003) Introduction to percolation theory. Taylor & Francis, Philadelphia, PA
6. Swanson LW (2012) Brain architecture: understanding the basic plan. Oxford University Press, New York, NY
7. Brodmann K (1909) Vergleichende Lokalisationslehre der Grosshirnrinde in ihren Prinzipien dargestellt auf Grund des Zellenbaues. Barth, Leipzig
8. Zamora-López G, Zhou C, Kurths J (2010) Cortical hubs form a module for multisensory integration on top of the hierarchy of cortical networks. Frontiers in Neuroinformatics 4(1):1–1
9. Richards W (2015) Anigrafs: experiments in cooperative cognitive architecture. MIT Press, Cambridge, MA
10. Richards W, Jepson A What is a Percept? University of Toronto, Dept. Computer Science, Tech. Report RBCV-TR-93-43
11. Hawkins J, Blakeslee S (2007) On intelligence. Macmillan, London
12. Mountcastle VB (1997) The columnar organization of the neocortex. Brain 120(4):701–722
13. Ballard DH (2015) Brain computation as hierarchical abstraction. MIT Press, Cambridge, MA
14. Moses J (2009) The anatomy of large scale systems revisited. Available from: http://esd.mit.edu/symp09/submitted-papers/moses-paper.pdf
15. Gick ML, Holyoak KJ (1983) Schema induction and analogical transfer. Cogn Psychol 15 (1):1–38
16. Spiro RJ, Vispoel WP, Schmitz JG, Samarapungavan A, Boerger AE, Britton BK, et al. (1987) Cognitive flexibility and transfer in complex content domains. Executive control processes in reading pp 177–199
17. Johnson TA, Caldwell BW, Green MG (2015) Investigating spontaneous flexibility in concept generation. In: ASME 2015 international design engineering technical conferences and computers and information in engineering conference. American Society of Mechanical Engineers, pp V007T06A012–V007T06A012
18. Tinbergen N (1951) The study of instinct. Oxford Clarendon Press
19. De Weck OL, Roos D, Magee CL (2011) Engineering systems: meeting human needs in a complex technological world. MIT Press, London

Chapter 42
Multiobjective Optimization of Geosynchronous Earth Orbit Space Situational Awareness Systems via Parallel Executable Architectures

Jordan Stern, Steven Wachtel, John Colombi, David Meyer, and Richard Cobb

Abstract This research implements a genetic algorithm (GA) to optimize geosynchronous Earth orbit (GEO) space situational awareness (SSA) systems via parallel evaluation of executable architectures on a high-performance computer (HPC). This effort has two main goals. The first is to develop and validate a methodology for optimization of large systems-of-systems in a robust manner that limits the assumptions typically necessary while performing large architecture trade studies. The second goal is to determine the set of near-optimal solutions for a GEO SSA system across multiple objectives. The GA is implemented in Python, and architectures are modeled and simulated with AGI's Systems Tool Kit (STK)™ on an Air Force Research Laboratory HPC. Results show how the GA finds increasingly "good" solutions, where the multiple objectives can be weighted and filtered. After 319,968 architectures were modeled, simulated, and evaluated on the HPC (27 years of CPU time, 3 days clock time), the near-optimal solution consisted of 10 globally distributed 1-m telescopes, 4 satellites in 1000 km equatorial low Earth orbit with 30-cm sensor apertures, and 3 satellites in GEO with 45-cm sensor apertures.

Keywords Model-based systems engineering • System architecture and complexity • Trade-space visualization and analysis • Systems engineering and decision science • Space situational awareness • Modeling and simulation • Optimization • Genetic algorithm

J. Stern (✉) • S. Wachtel • J. Colombi • D. Meyer • R. Cobb
Air Force Institute of Technology, Wright-Patterson AFB, Dayton, OH, USA
e-mail: jordan.stern@afit.edu; steven.wachtel@afit.edu; john.colombi@afit.edu; david.meyer.ctr@afit.edu; richard.cobb@afit.edu

42.1 Introduction

The growth of global economies has increased the number of nations with access to space, leading to increased dependence of many nations and businesses on space. Space assets play a critical role in modern militaries including the USA. The USA relies on space assets for positioning, navigation, and timing (PNT); intelligence, surveillance, and reconnaissance (ISR); early warning; and a host of other applications. Orbital regimes for Earth orbits can be divided broadly into four major categories: low Earth orbit (LEO), medium Earth orbit (MEO), highly elliptical orbit (HEO), and geosynchronous Earth orbit (GEO). As the name implies, satellites in GEO rotate round the Earth in the same amount of time that it takes the Earth to rotate once. Satellites in GEO appear to "hang" over one region of the globe, making GEO an ideal location for many communication and remote-sensing missions. As a result, many satellites of great economic and military importance are found in GEO.

Preventing accidental collisions and/or detecting other anomalous behavior is critical to maintaining the utility of space assets. The gathering of information about the current space environment to allow continued space operations in the face of unknown environmental conditions and the negligent and/or aggressive actions is called Space Situational Awareness (SSA). SSA functions are often divided into detect, track, and identification (ID). GEO SSA systems which perform detect and track functions are the focus of this research.

In this paper, the non-dominated sorting genetic algorithm II (NSGA-II) is used to optimize GEO SSA architectures by parallel simulation and evaluation of thousands of candidate architectures using a toolset composed of Python modules, AGI's Systems Tool Kit (STK)™, and Garrett's "inspyred" implementation of the NSGA-II evolutionary algorithm [1]. Evaluation of the architectures is based on each system's affordability and ability to detect and track resident space objects (RSOs) in GEO. The affordability and detection/tracking capability of each architectural candidate is collectively referred to as "fitness."

42.2 Background

42.2.1 Space Surveillance Network

The responsibility for performing many US SSA functions currently falls on the space surveillance network (SSN). The SSN consists of several assets that collect information on all orbital regimes. The assets dedicated to performing the GEO SSA mission include nine ground-based electro-optical deep space surveillance (GEODSS) telescopes, one space surveillance telescope (SST), one space-based space surveillance (SBSS) satellite, and four geosynchronous space situational awareness program (GSSAP) satellites [2, 3].

GEODSS sites are each equipped with three 1-m aperture telescopes and are located in Maui, Hawaii; Diego Garcia; and Socorro, New Mexico. Each GEODSS telescope is capable of making 575 observations in 1 h and can detect objects smaller than 40 cm in GEO [4]. SST is a single, prototype, 3.5-m aperture telescope located in Western Australia. While detailed performance data is not available, SST should be capable of making over 1000 observations per hour and detecting objects smaller than 10 cm in GEO [5]. While advances in daytime imaging are being made, both GEODSS and SST are currently limited to night-time observations and are impacted by local weather. This means that certain GEO RSOs can go unobserved for tens of hours or more [3].

SBSS is a LEO sun-synchronous orbit (SSO) satellite equipped with a 30-cm aperture telescope. LEO-based systems have a distinct advantage over ground-based systems because they are not subject to weather and have line-of-sight to every object in GEO every 90–120 min. LEO satellites, however, are limited from making observations of GEO RSOs with large phase angles. The phase angle is the angle between the RSO-observer vector and RSO-sun vector. At large phase angles, little of the light incident on the RSO from the Sun is reflected toward the observer. While precise performance data is not available, SBSS should be capable of 12,000 observations per day and detection of RSOs smaller than 100 cm [3].

Little is openly published about GSSAP. However, a system in a near-GEO orbit, like GSSAP, could potentially look out across GEO to overcome solar exclusion coverage gaps left by ground-based and LEO-based systems [3].

Although it has not yet been fielded, operationally responsive space (ORS)-5 is a program to develop an inexpensive LEO-based GEO SSA system. ORS-5 differs from SBSS in a way that it is an equatorial LEO orbit, co-planar with many satellites in GEO. This makes the relative angular velocity of the ORS-5 satellite to the GEO RSOs much smaller than it is at any time for a sun-synchronous LEO satellite (like SBSS). This, in turn, enables longer integration time and increased sensitivity at a given aperture diameter, ultimately leading to a smaller and less expensive satellite [3]. Although yet unproven, the ORS-5 concept is an interesting one and is also investigated in this paper as a part of the architecture.

42.2.2 RSO Detection

Detection of an RSO is the ability to distinguish the RSO from its background. Detection requires a sufficiently high signal-to-noise ratio (SNR) for the method used. An SNR of 6 is used as the detection criteria in this research work [3, 6]. Signal is determined by four major factors: RSO illumination, reflection, attenuation, and sensor efficiency. The signal is characterized by Eq. 42.1 as follows:

$$\text{signal} = N_e = \frac{P_{\text{rcvd}} \cdot \eta \cdot t_{\text{int}} \cdot \lambda_{\text{avg}}}{h \cdot c} \tag{42.1}$$

where:

P_{rcvd} is power received by the sensor (a function of RSO visible solar irradiance, RSO phase reflectance, RSO size, the distance between the RSO and sensor, atmospheric attenuation, optics attenuation, and the area of the collecting sensor).
η is the sensor quantum efficiency.
t_{int} is the sensor integration time.
λ_{avg} is the average wavelength of reflected light incident on the RSO.
h is the Planck constant.
c is the speed of light.

The signal is measured in electron count. The noise is signal uncertainty, assumed to follow a Poisson distribution, which comes from RSO signal, background signal, and sensor electronics. Sensor electronics noise includes dark noise and read noise. For ground-based systems, the background signal (S_e) includes terrestrially produced light, airglow, zodiacal light, star light, galactic light, and scattered moonlight [7]. For space-based sensors, S_e includes zodiacal light, star light, and galactic light. All background signals are assumed to be spread evenly across the sensor image. The noise is characterized by Eq. 42.2 as follows:

$$\text{noise} = \sqrt{N_e + n_{\text{pix}} \cdot S_e \cdot t_{\text{int}}} + n_d + n_r \tag{42.2}$$

where:

n_{pix} is the number of pixels on which the RSO signal falls.
n_d is the dark noise (a function of t_{int} and n_{pix} focal plane array design and temperature).
n_r is the read noise (a function of n_{pix} and focal plane array design) .

42.2.3 RSO Tracking

Determining the "goodness" of a GEO SSA system's ability to track an RSO requires knowledge of the orbit-determination accuracy that can be produced from the observations made by the system. Generally speaking, more observations are better than less, and evenly spaced (with respect to time) observations are better than tightly grouped ones [8]. Consequently, a particular architecture's tracking ability is measured using the mean of the maximum time gap between observations (natural proxy) made on each RSO in this work. The time-gap metric is referred to as "latency" throughout this paper.

42.2.4 System Cost

System-cost estimates were produced using cost-estimating relationships (CERs) for ground-based telescopes based on the work of Belle, Meinel, and Meinel [9] and for space-based telescopes based on the work of Stahl et al. [10]. The CERs use telescope aperture diameters *D(m)* (the inputs described by Eqs. 42.3 and 42.4) for both the ground-based and space-based telescopes:

$$C_{\text{obstry}} = \$4.0M \cdot D(m)^{2.45} \tag{42.3}$$

$$C_{\text{sat}} = \$400M \cdot D(m) \tag{42.4}$$

Operations and sustainment cost (O&S) estimates for telescopes were calculated by assuming that annual O&S cost was 20% of procurement cost as shown in Eq. 42.5. Constellation O&S costs were calculated using the AIAA space operations and support technical committee “complex mission” staffing levels with 50% overhead for ground equipment maintenance as shown in Eq. 42.6. A ten-year time horizon was used for the O&S estimates. Launch costs were determined using a launch vehicle selector written by the authors using estimated launch costs:

$$C_{\text{obstryop}} = C_{\text{obstry}} \cdot 0.20 \tag{42.5}$$

$$C_{\text{satop}} = \$9.9M \cdot \text{numCon} \tag{42.6}$$

where:

numCon is the number of unique constellations (LEO SSO, LEO Eq, and near-GEO).

42.3 Methodology

42.3.1 Multiobjective Optimization

Multiobjective optimization problems often require the search of a vast, complex solution space with conflicting objectives. A common approach is to reduce all of the objective functions into a single, normalized objective function based on the decision maker’s (DM’s) preference toward each objective. However, this technique provides the DM with a single “best” solution. A true multiobjective approach provides the DM with a set of solutions from which he/she can a posteriori choose the solution to implement.

While multiobjective optimization is often more computationally expensive than single-objective optimization, it provides the DM with a complete picture of the solution space and prevents expensive and time-consuming duplication of effort should the DM or his/her preferences change [11].

Genetic algorithms (GAs) are powerful tools for solving complex optimization problems. GAs operate on the following principles:

- Initial population is generated.
- Individuals are evaluated for "fitness."
- Individuals are chosen for "mating," with "fitter" individuals having higher chance of selection.
- "Parent" individuals pass on genetic makeup to "child" individuals.
- Random "mutations" are introduced in some children's genetic makeup.
- Children are evaluated for fitness.
- Process is repeated until termination criteria are met.

Deb et al. developed a GA, called non-dominated sorting genetic algorithm II (NSGA-II), specifically for solving multiobjective optimization problems [12]. This algorithm gives each individual a "non-dominance" rank in a population. Individual A dominates Individual B if Individual A is better than or equal to Individual B in all objectives, and strictly better than Individual B in at least one objective. Binary tournament selection is used to select mating pairs. Two individuals are chosen at random, and the one with the better non-dominance rank is allowed to reproduce. If the two individuals have the same non-dominance rank, the selected individual is the one with the fitness score of a larger Euclidian distance from other individuals. This maximizes the diversity of the candidate solutions [12]. Three objective functions are considered for this paper:

- Minimization of detectable size of GEO RSOs
- Minimization of maximum time gap between observations of each GEO RSO
- Minimization of system cost

This research work uses NSGA-II as an optimizer in conjunction with an executable architecture, in the form of a script-driven physics-based simulation with automated post-processing. The following sections and diagrams describe the process developed and used in the work.

42.3.2 Architecture

The present research work models the system architecture (individual) as a Python list. Each element (gene) of the list represents a design variable in the architecture. Table 42.1 displays the architectural variables and their possible ranges.

42.3.3 Architecture Generator

The generator randomly creates an initial population of feasible architectures (individuals). This initial population size is set by the user. A standard initial

Table 42.1 Architectural parameters and ranges

Architectural parameters (genes)	Lower bound	Upper bound	Step size
# Ground telescopes (at each of nine locations)	0	4	1
Ground telescope aperture diameter	0.5 (m)	4	0.5
LEO sun-synchronous altitude	500 (km)	1000	100
LEO sun-synchronous satellites per plane	0	2	1
LEO sun-synchronous planes	1	2	1
LEO sun-synchronous aperture diameter	0.15 (m)	1	Varies
LEO equatorial altitude	500 (km)	1000	100
LEO equatorial number of satellites	0	4	1
LEO equatorial aperture diameter	0.15 (m)	1	Varies
Near-GEO observer altitude (Δ from GEO)	−1000 (km)	1000	500
Near-GEO observer number	0	4	1
Near-GEO observer aperture diameter	0.15 (m)	1	Varies

population size of 96 individuals was selected by the researchers for this work, based on high coverage of all possible gene values (“alleles”). Each individual contains 28 genes, with a maximum alphabet size of eight. This means that an initial population of 96 individuals results in 99.9% probability of every allele being present in the population [13]. This high allele coverage gives the GA a higher probability of convergence. Once all individuals in the initial population have been generated, the population is sent to the evaluator.

42.3.4 Architecture Evaluator

Upon receiving the list of candidates from the generator, the evaluator distributes each candidate to a “sub-evaluator” that is responsible for the modeling, simulation, and fitness determination of that single architecture. Sub-evaluations occur in parallel. At each instance, the sub-evaluator creates a list of commands that implement the architecture it was given and sends those commands to STK. STK then models and simulates the architecture and produces reports that contain the following information:

- Time periods of line-of-sight access between each RSO/sensor platform pair
- Range between each RSO/sensor pair during periods of access, at a 30-s time step
- Phase angle for each RSO/sensor pair, at a 30-s time step
- RSO zenith angle for each RSO/ground sensor pair, at a 30-s time step
- Lunar zenith angle for each RSO/ground sensor pair, at a 30-s time step
- RSO-Lunar separation angle for each RSO/ground sensor pair, at a 30-s time step
- Lunar phase

Upon completion of the STK simulation, the reports are parsed and a sensor-tasking schedule is created. The schedule creation is necessary because STK has no built-in Linux-compatible capability for creating sensor-tasking schedules. The primary issue this creates is that STK will point a sensor at an RSO as long as it has line-of-sight access. This issue was handled with a simple scheduler built in Python. The scheduler generates a list of possible access intervals for the scenario, a "counter" which keeps track of how many 30-s intervals have elapsed since each RSO was last observed, and a list of possible accesses for each RSO/sensor platform pair. At each time step and for each sensor, the accessible RSO with the highest counter value is selected for observation. This is essentially an implementation of the "greedy" heuristic. The completed schedule is used directly to determine latency between observations for each RSO. It is also used to determine which values to pull from the other STK reports to determine minimum detectable RSO size for each observation.

The mean of the minimum detectable RSO size, the mean of the maximum observation time gap for each RSO (i.e., latency), and system life cycle cost (determined directly from the architecture, no STK required) are, collectively, the fitness of an architecture.

Upon completion of the evaluation of an architecture in the population, its fitness is used to sort architectures by dominance. Selection, crossover, and mutation operations are performed to generate a new population, and the process is repeated. A running archive of all non-dominated architectures and their fitness is compiled for the entire process. Figure 42.1 depicts the high-level process described in the preceding paragraphs.

42.4 Results

42.4.1 Genetic Algorithm Validation for Architecture Optimization

The following results are from the synthesis of 32 trials, each with a population size of 96, over 100 generations, and a 5% mutation rate. RSOs were generated from all 813 two-line element sets (as of the writing of this paper) in the GEO RSO catalog (http://space-track.org). Trials were simulated for the Northern Hemisphere summer solstice and vernal equinox. A total of 319,968 architectures were modeled, simulated, and evaluated. Each trial took approximately 3 days (executed in parallel), equating to 27 years of CPU time.

A plot of best, median, and mean values over successive generations shows over 50% improvement in the solution values in the first 10 generations, as shown in Fig. 42.2. This is a strong indication that the algorithm is working. All single attribute value functions were linear. Bounds used are given in Table 42.2.

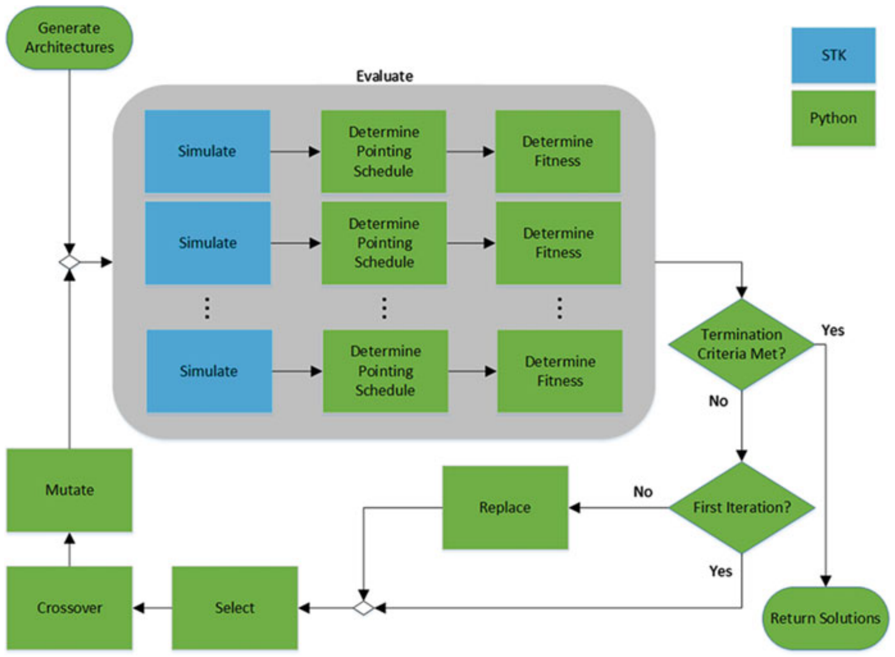


Fig. 42.1 High-level methodology for parallel physics-based simulations within a genetic algorithm

Fig. 42.2 Value growth

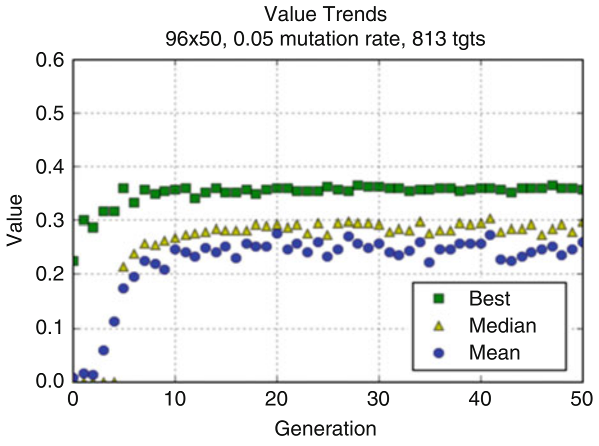


Table 42.2 Value function bounds

Latency	>90 min, value = 0	0 min, value = 1
Size	>0.75 m, value = 0	0 m, value = 1
Cost	>$3B, value = 0	$0, value = 1

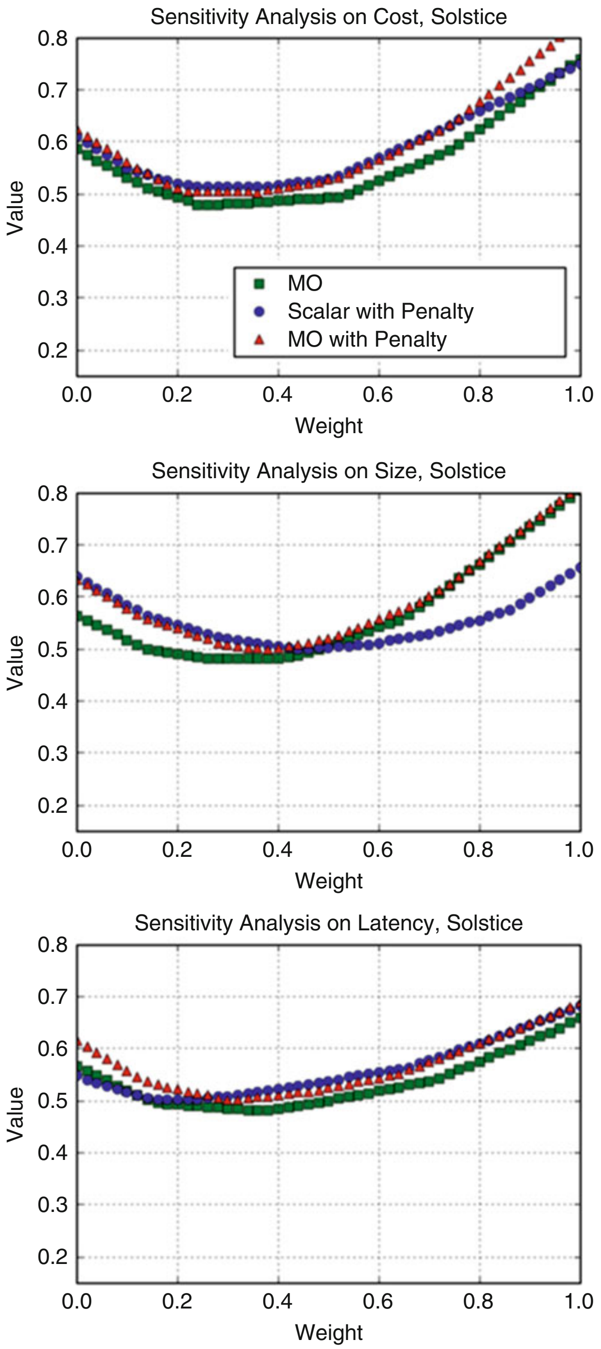


Fig. 42.3 Sensitivity to objective function weight

Three fitness evaluation methods were explored: multiobjective (MO) with unbounded solution space, MO with a penalty function, and scalar with a penalty function. The penalty functions ensured that designs with fitness – in any of the three objectives – outside the defined bounds were given infinitely poor scores, pushing the GA toward reasonable solutions. Equal weights were used for the scalar objective function, which is defined in Eq. 42.7. The scalar approach effectively transforms the problem into a single-objective problem; however, it requires preference weights to be assigned prior to the analysis.

Figure 42.3 displays the sensitivity analysis, examining the effect of objective weighting on the performance of each evaluation type. Trial results were screened using the aforementioned bounds. For clarity, the value axis is considered from 0.15 to 0.8. The sensitivity analysis shows that a scalar evaluation of multiple fitness measures will produce results as good as or better than MO under most circumstances. A notable exception in these results is the performance of the scalar evaluations when the weight on size is high. The authors believe that this is because the bound on the size measure was too relaxed. MO never performed best, even at extreme weights. MO-penalty performed nearly as good as scalar with penalty and even better in some cases.

For an analysis where the solution space is not well understood, re-analysis is undesirable, or preferences on the objectives are unknown, MO with a penalty function that implements reasonable bounds is likely to be the best method for a complex optimization problem.

42.4.2 Near-Optimal Architecture Results: Value

The best performing architectures from the solstice were run on the equinox and vice versa. The solstice architectures performed better on the equinox than the equinox architectures did on the solstice. Therefore, the results for the four scalar trials simulated on the summer solstice are presented in more detail below. The four best architectures can be divided into two families distinguished by the space-based portion of the architecture.

Table 42.3 lists the significant parameters of the space-based portion of the architectures, as well as the average performance of each family across the three objectives. The differences between the ground system architectures were less significant and had less influence on the overall system performance.

All trials yielded solutions with several telescopes at La Palma and the Indian Astronomical Observatory (IAO), and either Mauna Kea or Haleakala, HI. This is a logical result, considering that these locations are fairly evenly distributed around the globe, providing coverage of most of the GEO belt each night:

Table 42.3 Architecture families and their fitness

	Family 1	Family 2
Architecture parameter		
SSO Sats	0	0
Eq. LEO Sats	4	4
Eq. LEO Sat Ap. Dia.	0.30 m	0.30
Near GEO Sats	4	3
Near GEO Sat Ap. Dia.	0.30 m	0.45 m
Fitness		
Size (cm)	44	34
Latency (min)	35	46
Cost ($B)	1.44	1.47
Value	0.513	0.512

$$f_{\text{obj}}(X) = \frac{1}{3} * u_1(X) + \frac{1}{3} * u_2(X) + \frac{1}{3} * u_3(X) \tag{42.7}$$

where:

$f_{\text{obj}}(X)$= 1 – value, the new objective function

$$u_i(X) = \begin{cases} f_i(X) \text{ } \mathit{if\ all} \, f_i(X) \leq Bound_i \\ 1 \text{ } \mathit{if\ any} \, f_i(X) > Bound_i \end{cases} \text{for } 1 \leq i \leq 3$$

$f_1(X)$,$f_2(X)$,$f_3(X)$ are the original Size, Latency, and Cost objective functions

$$X = \{x_1, \ldots, x_{28} | \text{possible allele values}\}$$

Paranal, Chile, while only selected by half the trials, was preferred over Socorro, NM, and Mt. Graham, AZ. It covers a similar longitude and experiences longer nights on the summer solstice.

As mentioned before, the solstice trial results were simulated on the equinox to determine their performance with different illumination conditions and the eclipse. Evidently, Family 2 architectures performed better than Family 1 on the equinox because they maintained most of their size advantages, and the eclipse reduced the latency advantage of the Family 1 architectures (see Table 42.3). As a result, the nearest-to-optimal solution at the given weights and bounds is comprised of a "Family 2" space architecture, three 1-m telescopes at Mauna Kea and La Palma, and four 1-m telescopes at IAO.

42.4.3 Near-Optimal Architecture Results: Performance

As a matter of practice, most DoD programs seek to maximize performance within a given fixed budget and schedule. Along similar lines, the results were analyzed

Table 42.4 Space architecture and fitness, max performance trials

Architecture parameter	
SSO Sats	2*
SSO Sat Ap. Dia.	0.45 m**
Eq. LEO Sats	4
Eq. LEO Sat Ap. Dia.	0.45 m
Near GEO Sats	4
Near GEO Sat Ap. Dia.	0.45–0.75 m
Fitness	
Size (cm)	23–28
Latency (min)	35–41
Cost ($B)	2.83–2.99
Value	0.611–0.623***

with bound cost (max of $3B) and objective function weight shared equally between size and latency. In this case, the scalar trials no longer produce the best results and are edged out slightly by the MO with penalty trials. Among the four highest-performing MO with penalty trials and new weights, there were again significant architectural similarities. Among the space portions of the architectures, there were a set of parameters that were common in all the trials, with two trials manifesting several differences. Table 42.4 lists the common space architecture along with the performance ranges for all four trials. It is the opinion of the authors that the SSO satellites would not have been selected and had the model permitted higher numbers of equatorial LEO and/or near-GEO satellites. The selected high-performance architecture was the one that overall performed best on both the solstice and equinox. It is described in Table 42.5.

There was significant variance among ground-based telescopes with as few as two telescopes in one architecture and a maximum of 11. Aperture diameters ranged from 0.5 to 1.5 m. The small numbers of telescopes are likely to be the results of the way the size metric was implemented. Because space-based systems make many more observations, their detection capabilities act as a "lowest common denominator." There is little incentive in the model to make a small number of high-sensitivity observations.

The preferred near-GEO constellations placed satellites in super-GEO. Sensitivity to near-GEO altitude increased as the number of near-GEO satellites decreased. This is because the solar exclusion gap is easily closed by three or four near-GEO satellites regardless of exact altitude, whereas two-satellite constellations must be in super-GEO to close the gap. LEO satellite altitude was less stable, likely because it has less influence on performance.

Table 42.5 Near-optimal architecture summary, solstice performance

Design goal	Parameters	La Palma#, size (m)	Haleakala	M. Kea	IAO	SSO#, size (m), Alt (km)	LEO Eq.#, size (m), Alt (km)	GEO#, size (m), ΔGEO Alt (km)	Size (cm)	Latency (min)	Cost ($B)
Max value	3,1	0	3, 1	4, 1	0	4, 0.3, 1 k	3, 0.45, +0 k	34	46	1.47	
Max performance	4,1	2,1.5	1, 0.5	4, 1.0	2, 0.45, 0.7 k	4, 0.45, 0.7 k	4, 0.75, +1 k	27	35	2.99	

42.4.4 Interpretation

Some clear methodological themes emerged in the course of this work. First, the use of a GA was shown to be a good method for tackling the problem. Second, the MO-penalty evaluation method emerged as a good compromise between unconstrained multiobjective and scalar evaluation while avoiding the pitfalls of each – searching undesirable solution space and being inflexible to changing requirements, respectively.

There are a few more key takeaways from an architectural perspective. First, the robustness of super-GEO satellite architecture should be the backbone of any future GEO SSA system. This is the result of the super-GEO satellites' wide field of regard and their ability, with as few as two satellites, to detect RSOs that would normally be in solar exclusion. Second, the ORS-5 concept is clearly a good one. It appears that the increased performance per dollar is worth the high delta-V required to get to equatorial LEO from US launch locations, so much so that none of the high-value architectures included SSO satellites. Lastly, despite the size-metric sensitivity issue discussed above, the highest performing architectures included relatively large numbers of telescopes. This confirms that ground-based telescopes should continue to play a large role in GEO SSA, even with advances in space-based systems.

42.5 Conclusion

This paper has shown that a GA can be used in conjunction with large, externally controlled simulations to find near-optimal solutions for a system with a very large trade space. Parallelization and automation of tasks that are otherwise normally done sequentially through a GUI (like running STK) enables rapid evaluation of many architectures and increases the chance of finding "the solution" in a reasonable time frame. The rigorous nature of the modeling and simulation technique (which modeled every observation made by each system, over a 24-h period, on the full set of published GEO RSOs), coupled with a robust optimization methodology, goes far beyond what has been done to date. As a result, this work also serves to verify many of the GEO SSA architecture point designs developed over the last several years. Of particular note is the fact that model development through final results took only a few months, achieving many lifetimes of point design evaluations. Unfortunately, point design evaluation is still how much of the trade space analysis is done in the DoD.

There is still much that can be done to improve the model. Proposed follow-up work is to examine the assignment of sizes to each RSO, which will enable wrapping the size and latency objectives into one. This is expected to increase sensitivity to telescope size and location and to provide a more realistic representation of the SSN behavior. The model also allows simple integration of additional SSA objectives (such as ID and characterization, sparse observation tracking, and high-priority RSO custody) which should be examined in future implementations. More robust weather and cost modeling, along with longer scenarios, would also improve the fidelity of the results. Finally, the fact that the best solutions found were at the maximum of many of the system boundaries indicates that better solutions may exist beyond these boundaries which would be explored in future work.

The paper demonstrates true executable architecture MBSE that could be easily applied to other aerospace problems with some modifications. It also produced a range of preference agnostic, near-optimal solutions to the GEO SSA problem, an outcome which could otherwise not have been obtained in a lifetime using the current point design approaches.

Disclaimer The views expressed in this paper are those of the authors and do not reflect the official policy or position of the US Air Force, Department of Defense, or the US Government.

References

1. Garrett A “inspyred,” python. Python Software Foundation, 25-Jul-2015
2. SMC PA, Air Force successfully launches GSSAP 3/4 – AFSPC-6 from Cape Canaveral, Air Force Space Command, 19-Aug-2016. [Online]. Available: http://www.afspc.af.mil/News/Article-Display/Article/920089/air-force-successfully-launches-gssap-34-afspc-6-from-cape-canaveral-afs. Accessed: 09 Sep 2016
3. Ackermann MR, Kiziah RR, Zimmer PC, McGraw JT, Cox DD (2015) A systematic examination of ground-based and space-based approaches to optical detection. In: Proceedings from 31st Space Symposium
4. Bruck RF, Copley RH (2014) GEODSS present configuration and potential. In: Proceedings from Advanced Maui Optical and Space Surveillance Technologies Conference
5. Shah R, Woods DF, Faccenda W, Johnson J, Lambour R, Pearce EC, Stuart JS (2013) Asteroid detection with the space surveillance telescope. In: Proceedings from Advanced Maui Optical and Space Surveillance Technologies Conference
6. Koblick D, Goldsmith A, Klug M, Mangus P, Flewelling B, Jah M, Shanks J, Piña R, Stauch J, Baldwin J, Campbell J, Blake T (2014) Ground optical signal processing architecture for contributing space-based SSA sensor data. In: Proceedings from Advanced Maui Optical and Space Surveillance Technologies Conference
7. Fruh C, Jah MK (2013) Detection probability of earth orbiting objects using optical. In: Proceedings from AAS/AIAA astrodynamics specialist conference, pp 3–14
8. Horwood JT, Poore AB, Alfriend KT (2011) Orbit determination and data fusion in GEO. In: Advanced Maui Optical and Space Surveillance Technologies Conference, p 7
9. van Belle GT, Meinel AB, Meinel MP (2004) The scaling relationship between telescope cost and aperture size for very large telescopes. In: Astronomical Telescopes and Instrumentation
10. Stahl HP, Henrichs T, Luedtke A, West M (2011) Update on parametric cost models for space telescopes. In: SPIE UV/Optical/IR Space Telescope and Instruments

11. Evans H, Lange J, Schmitz J (2015) The phenomenology of intelligence-focused remote sensing, Volume 1: electro-optical remote sensing, vol 1, 1st edn. Riverside Research, New york
12. Deb K, Pratap A, Agarwal S, Meyarivan T (2002) A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans Evol Comput 6(2):182–197
13. Reeves CR (1993) Using genetic algorithms with small populations. In: Proceedings from 5th international conference on genetic algorithms

Chapter 43
System User Pathways to Change

Lt Col Amy Cox and Zoe Szajnfarber

Abstract This study was motivated by observations of user innovation within a study of functional gains to a complex system; the user changes introduced novel function with a seemingly minimal reliance on material change. To better understand how users were realizing these changes, empirical research of user change behaviors was accomplished. This present research unpacks user change behaviors through an inductive analysis of four cases of user design. Although material changes were observed, they were not a necessary condition for functional gain; rather, system users demonstrated a reliance upon the introduction of novel system configurations (operational change) and novel user task structures (human change). This paper presents this inductive study, considers the pathways for change employed by system users, and motivates future research for leveraging these pathways for change.

Keywords User innovation • Flexibility • Changeability • Design

43.1 Introduction

Literature on flexibility and changeability has posited theory on how to design systems to ease the gain and change of functionality over time [1–3]. This literature has considered design choices that can reduce the magnitude or cost of late changes to the form of a system. The empirical literature in this space has focused on issues related to changing form. Implicit in this stream is the notion that functional gains require a commensurate change to form. Several theoretical perspectives have also suggested that low-cost flexibility can be achieved without form changes. However, there is a limited empirical basis for this perspective, and as a result, the specific

L.C.A. Cox, PhD (✉)
Air Force Institute of Technology, Wright-Patterson AFB, OH, USA
e-mail: amy.cox@afit.edu

Z. Szajnfarber, PhD
The George Washington University, Washington, DC, USA
e-mail: zszajnfa@gwu.edu

mechanisms to achieve constant-form functional gains have not been articulated. In this work, we add empirically derived flexibility mechanisms to that discussion.

The class of flexibility mechanism articulated herein is associated with function-expanding changes that users make naturally. This present research was motivated by the empirical observation of system users introducing novel functions to complex systems in postproduction [4]. In line with the definition of flexibility, the changes involved minimal changes to the form of the baseline system (ease), yet provided completely novel and significant functional gain. In this work, the process through which users made these changes – both in terms of the system attributes that enabled them and the particular levers they chose to turn – is examined in detail. The basis for this study is a richly documented large-scale unconventional military operation that was known to rely on several functional gains to existing material solutions. The intent is that by identifying and explaining the mechanisms exploited by system users, novel pathways to form-free functional gains (in general) will be revealed.

43.2 Relevant Research

This research is motivated by a challenge faced by the U.S. Air Force, managing aircraft fleets to meet the changing needs. The needs for a given class of aircraft shift over time; enemies adapt, technologies emerge, and new missions are envisioned. Historically, the challenge of providing material solutions for changing needs was met through aircraft development and replacement [5]. At present, replacement is no longer apace with functional change; development alone can take from 10 to 15 years [5]. In place of replacement, reliance on modification programs has grown [6]. We can also see the advent of policies for open system architecture [7] and practices, which improve customer leverage over a system's life cycle [8]. The challenge of changing needs has been met with an effort to facilitate material change.

Not surprisingly, the specific challenges we see with the Air Force are areas of interest within the broader systems engineering literature. Through abstracting the specific problem, we see the more general challenge of coping with requirements uncertainty (changing needs) with complex engineering systems (aircraft fleets) [1]. The literature has theoretically explored design strategies, which largely fall into either passive (robust design) or active (flexible design) measures [1, 9]. Other delineations consider whether the system is changed externally, introducing concepts of systems designed to adapt themselves (internal change) and systems that are easy (flexible) and fast (agile) to change (external change) [2, 10]. Unfortunately, these tactics come at a cost; thus, there is literature on when these aspects are necessary [2, 11] and how to consider the trades between optimal and over design [12].

Within the literature, the discussions often cite the need for more empirical study [2, 3, 13]. Along these lines, in recent research, the authors studied instances of a complex system realizing functional gains over 30 years of its postproduction life cycle [4]. We observed empirical cases where passive (e.g. robust design margins) and active (e.g. modular interfaces) design aspects facilitated functional gains. An interesting phenomenon emerged when the actors in these changes were considered. Among our cases, we observed instances where system users were both the source of novel ideas and the integrators of novel functionality. The presence of user innovation in this environment is interesting because it has implications for both flexible design as well as for the qualitative type of changes introduced in postproduction.

User innovation is a concept from management and marketing literature. The term user is linked to one's functional relationship to a system. A user is one who benefits through the use of something [14]. This definition can be broad, extending from a parent who fabricates the first ever jogging stroller in their garage [15] to a large firm that creates scientific instrumentation to meet their specific research goals [16]. To narrow this perspective, our observation was comparable to the former, we observed innovations driven by a limited group of aircrew members and not the Air Force at large.

In light of the category of user, the presence of user innovation in this setting points to the possibility of a low barrier to change. Users benefit from an asymmetric understanding of their needs relative to manufacturers [17, 18]; they can have unrivaled functional expertise and draw upon a different base of knowledge when they innovate [19]. However, as systems grow in complexity, functional expertise is not necessarily enough; the knowledge required to innovate grows and other expertise must be leveraged to realize change [18]. In addition to barriers due to knowledge, innovation theory posits that single user innovation is most viable when design cost is low [20]. The presence of single user-driven change to a complex system points to change with a relatively narrow base of expertise and a low cost to implement.

The presence of this category of innovator also points to a source of qualitatively different changes. The innovation literature has found that users tend to introduce completely new dimensions of performance [21, 22]. In contrast, when innovating, manufacturers tend to introduce efficiency and reliability-based innovations. Due to its demonstrated commercial potential, there has been substantial marketing and management research of user innovation [14, 21]. Yet, although the literature has unpacked when and why users innovate, the questions of how they design and realize changes have remained underexplored.

We started this review with our motivation, the challenge of complex systems that must remain functionally relevant in the face of uncertain requirements. Although the motivating problem was specific to the Air Force, there is an extensive and more general exploration of these concepts within the systems engineering literature. Next, returning to a more specific Air Force setting, we highlighted the presence of user innovation in a postproduction complex system. The presence of user innovation points to the possibility of novel functionality with a low barrier to change. Although the extant innovation literature has deeply explored the market concerns related to this phenomenon, users design and change methods remain underexplored. This present research seeks to understand how user innovators realize change in complex systems. This line of inquiry has been pursued as it may yield further insights into design to facilitate change.

43.3 Research Approach

To summarize the framing of this research, we are interested in understanding user design behaviors as they may yield insight into flexible system design. Existing studies of design behavior typically leverage protocol analysis of designers while they design in lab settings in response to notional problems [23–25]; these methods are not suitable for the phenomenon of interest. User innovation occurs when a user has a need that is not met by current systems [14]; users design to meet their local needs while employing their local methods [17]. The need-based emergence of user innovation does not make it amenable to prompted protocol studies.

Design literature has also employed retrospective studies of designer activities [26] and studies of designer output [27] to derive meaningful insight. Bearing these output-based studies in mind, the present research exploits a richly documented historic setting, which offers a range of cases of user design. Two levels of analysis were considered in this research: individual design decisions and entire designs. Analysis categories for individual change actions, the lowest level of analysis, were inductively developed from the data [28, 29]. This data-driven approach was selected as it permits a rich and accurate description of a given phenomenon when existing explanations are lacking [29]. This current work will focus on the categories of user change, which were observed.

43.3.1 Research Setting

The development effort associated with Operation Kingpin is the setting for this research. Operation Kingpin was an unprecedented unconventional warfare operation to rescue an estimated 60 Prisoners of War during the Vietnam War. Through a series of system changes, an air element of four different aircraft types (a total of 13 aircraft and 62 aircrews) were integrated together with a ground force of 56 Army Special Forces to provide a system capable of meeting unique mission requirements.

This setting was selected because it provides a critical case of user innovation. First, based on an initial review of historical studies, the urgency of the operation resulted in mission design, development, and integration, which was accomplished by the user community. Second, there is an a priori expectation of several design efforts and changes to existing material systems; the extant literature has demonstrated the presence of numerous innovations (i.e. the phenomenon sought) in the preparation for this mission. Third, there were a diversity of design efforts covering a range of system complexity, thus permitting a view of user innovation with complex systems.

Next, this research setting was selected due to the diversity and accessibility of available data. Due to the significance of this operation, both in the political reaction immediately after it occurred and in military strategy studies that followed, a wealth of official documentation [30], transcribed interviews from those directly involved, secondary history sources [31, 32], and related histories [33–36] exist. The data sources span multiple perspectives, providing diversity to support construct validity [29]. The understanding of this data was facilitated by the primary author's background testing military aircraft.

43.3.2 Case Selection

The core data for this research is the publicly released components of the Operation Kingpin After Action Report. The report was written by personnel who were part of the planning and execution of the mission, and it was completed within a month of the mission [30]. The report covers the 6-month window from early analysis and mission design and integration through mission execution and assessment. The final mission plan for Operation Kingpin also served as a data source. Although a multitude of documents were reviewed, the cases considered drew directly from 14 sources (3397 pages).

To gain familiarity with the data, multiple histories of this mission were reviewed. The After Action Report was used to identify instances of design efforts completed during the mission preparation. Cross comparisons were made between the histories and the After Action Report to discern the available cases and the amount of data for each. Table 43.1 lists the design cases that were selected and describes the case data. Four cases were selected according to the following criteria: (1) Consistent with the research setting selection, each case was an instance where system users led the design effort. (2) Adequate data was available to trace a sequence of design decisions and design changes. (3) Theoretical selection based on apparent complexity to permit the consideration of two cases each of low- and high-complexity designs (literal and theoretical replication). (4) The cases represented a diversity of technologies to avoid idiosyncratic findings due to specific technologies.

A database was created for each of the four cases. Entries within the database were inclusive of relevant historic information, problem framing, design choices, and implementation information. Entries were placed in chronological order to permit a viewing of the flow of the design activity. For each entry, excerpts from multiple sources were included when available. All excerpts were tied to their origin to allow for a chain of evidence [37].

Table 43.1 Case selected

Design	Relative complexity	Need	Change description	Changes observed
Machete	Low	The mission required the removal of prisoners from captivity without harming them. Interviews with former prisoners of war indicated that a variety of locks, chains, and other barriers could be expected in north Vietnamese prisons. The users needed to develop a means of defeating barriers without doing harm to prisoners.	Rescue personnel were selected for exceptional characteristics including mastery of specialized equipment. Through exercises with a mock prison, the need for a blade for hacking and prying was identified. A standard issue military machete was ground down to a modified profile (shorter and sharper) to address barriers that required prying. Fire axes, fire fighter's bolt cutters, and acetylene torches were used for other barrier types.	Material (3) Operational (0) Human (7)
Night sight	Low	The mission required the defeat of enemy personnel within the compound; guards needed to be neutralized before they could either harm the raiding team or the prisoners. This necessitated rapid close quarters operations within the prison compound and high-accuracy weapon fire (single shot on target) to avoid inadvertently shooting prisoners. The mission was planned for night time to increase the element of surprise.	Personnel were selected for exceptional characteristics including marksmanship. Mission timing was set to exploit natural moonlight conditions to improve visual conditions for the ground party. Following a transition from daylight to moonlight conditions in training, accuracy dropped to 35%. A low-light commercial scope was acquired and mounted on a military rifle using modified equipment and novel techniques. New firing techniques were developed, trained, and mastered to attain 95% accuracy.	Material (6) Operational (0) Human (15)
Navigation marker	Med-high	Attack aircraft, which protected the ground forces from incoming enemy reinforcements, required a navigation marker to stay in an exact location in	An existing napalm bomb was placed in a novel configuration with a custom wooden cradle and a surplus drag chute. This device was airdropped from the	Material (3) Operational (5) Human (4)

(continued)

Table 43.1 (continued)

Design	Relative complexity	Need	Change description	Changes observed
		proximity to the ground forces. Standard daytime visual references (e.g. road intersection) were not adequate at night. The users needed to develop a means to aid the navigation of the attack aircraft.	ramp of a cargo aircraft. The nonstandard low-speed vertical employment allowed the flammable liquid in the bomb to pool rather than disperse. This provided a 40-foot high flaming beacon that burned for an hour. Precision placement and timing of the airdrop required extensive planning and repetition.	
Low speed MC-130E	High	Small helicopters capable of safely, crash landing in the prison compound were not capable of precision navigation or night time operations. The prison was located deep within enemy territory in a hostile environment with substantial antiaircraft defenses. The mission was planned for the night to reduce enemy threats. The users needed to develop a means to precisely navigate the helicopters at night.	A cargo aircraft with a first generation precision navigation system, capable of flying below enemy radar detection, was selected as a pathfinder (MC-130E). Modular material changes were made to improve navigation and formation interaction. Novel configurations were developed to reduce airspeed to a helicopter-compatible speed and retain adequate flight control response. Procedures to surf the helicopters on the wing tip vortex of the MC-130E were developed to increase the helicopters' speed. Extensive cycles of training and mission planning were required.	Material (3) Operational (13) Human (14)

43.3.3 Analysis

The data analysis methods were based on existing methodology literature in process data, case study research, and qualitative analysis [28, 29, 37–39]. Due to the lack of existing research and theory on user design methods, a bottom-up inductive approach was selected to build theory. The previous sections explained the

Table 43.2 Change categories

Category	Data sample
Material	"For the raid, a GAR/I *beacon was installed* on each helicopter ... (Thigpen, p. 144)" (low speed, 13)
	"They helped Eglin carpenters to *design and build special wooden cradles* that allowed the talons to jettison the firebombs from the ramp. (Gargus, p. 59)" (marker, 5)
	"They asked the base machine shop to *alter standard government machetes* into the desired blade . . . (Schemmer, p. 113)" (machete, 12)
Operational	"The two aircraft commanders also found that by using the two inboard engines at high-power settings with the outboards at reduced power, the combat talon could fly at a lower airspeed than the computed stall speed. (Thigpen, p. 145)" (low speed, 21)
	"To establish the proper drafting position, the C-130 had to fly under the Huey, which had to drop down just aft of the left wing with its fuselage centered on the edge of the wing tip. (Gargus, p. 53)" (low speed, 27)
	"After dropping flares over the prison, this first C-130 makes a right turn and drops fire fight simulators... The C-130 then continues. . . and drops two firebombs, marker flares, and fire fight simulators... (JCS plan, p. 4)" (marker, 14)
Human	"Throttle *technique was the key* to slow-speed flight, which was *a skill the pilots developed* during the training phase (Thigpen, p. 171)." (low speed, 29)
	"Fifty-one primary personnel and ten backup personnel *were selected on the basis of their performance* during phase I training with special emphasis placed on leadership, MOS, rank, physical stamina, special skills, and other attributes. . . realizing the assault group was unique in its mission, *heavy emphasis was placed on the selection of the more experienced, skilled, mature personnel*. (JCS, Pt II, p. E-1)" (machete, 7)
	"These insertions were conducted during daylight and at "walk through" pace. *As proficiency increased*, the pace was stepped up until real-time movement was achieved. Simultaneously, training progressed from dry to live fire in the target area and the insertion of all ground elements in real-time sequence (JCS, Pt I, p. 22)." (sight, 33)

rationales behind the research method, the setting selection, and the methods to attain data. This section will provide a description of the analysis process.

The analysis is built on previous analysis of a different set of cases from this research setting [40]. The unit of analysis was individual change actions from user designers. The data for each change action were raw text excerpts from the historic sources; individual change actions were often supported with multiple historic sources. In cross-comparing these change actions, three change categories emerged based on what aspect of the system the users were changing. We found that the users were changing the material system (material change), how they used the system (operational change), and the human aspects of the system (human change). Table 43.2 provides excerpts of data from individual changes to illustrate these different categories of change. The within-quote annotation traces to the original data source, and annotation outside of the quote is to the specific case and change within that case.

Table 43.3 Characterization of material changes

Change description	Level	Traits
Use new Army machete with desired profile (as is). (machete, 10)	1	Select existing material to meet requirement
Use existing commercial machete with desired profile (as is). (machete, 11)		Select existing material to meet requirement
Use existing light weight rifle (GAU-5/5A). (sight, 3)		Select existing material to meet requirement
Use the MC-130E; aircraft has precision low level navigation. (low speed, 1)		Select existing material to meet requirement
Use tape to tighten scope attachment to rifle. (sight, 24)	2	Simple change. Common material. Reversible.
Attach infrared scope to existing rifle. (sight, 13)		Modular addition of component.
Attach commercial hunting scope to existing rifle. (sight, 15)		Modular addition of component.
Add existing sling to rifle. (sight, 32)		Modular addition of component.
Modify existing standard army machete by grinding the blade to desired profile. (machete, 12)		Simple change using basic technology (grinder). Permanent.
Notch available screwdriver to match custom interface. (sight, 26)		Simple change using basic technology. Permanent.
Attach wooden cradle to napalm bomb. (marker, 6)		Modular addition of component.
Attach existing chute to wooden cradle. (marker, 7)		Modular addition of component.
Attach existing GAR/I subsystem to aircraft (bolt on locally). (low speed, 13)		Modular addition of component.
Create a cradle from wood to encase the napalm bomb, enable connection to parachute and ease cargo handling. (marker, 5)	3	Creating a new component/interface using basic technology (wood working tools).
Attach existing FLIR subsystem to aircraft, aircraft sent to Lockheed facility for change. (low speed, 12)		Modular addition requiring specialized expertise.

A simple method of comparing the diverse changes to one another was desired; a method to permit both within and across category comparisons of user changes. Ultimately, the ability to code changes and compare them can facilitate pattern recognition with rich qualitative data. Four levels per category were chosen to enable some resolution while remaining parsimonious. Within each category, the changes were ordered from low to high, where low represents the use of something already in existence without change and high represents a change analogous to creating a multicomponent system. Clusters emerged in comparing the ordered changes, and the changes were placed into one of the four levels. Nominal definitions of each level were developed through considering the traits associated with the changes in that level. An example of this clustering can be seen in Table 43.3 for the material changes; it is important to note that no Level 4 (completely new

Table 43.4 Description of change categories and occurrence in data set

Level	Material	Operational	Human
1	*Use existing*. Seeking out and selecting an existing material solution, without modification, to meet emerging needs. *Example:* Determining that a need to cut chains/bonds of prisoners existed, identifying a commercial market machete capable of meeting that need and procuring that machete. *Occurrences*: 4	*Use existing*: Adopting established operational configurations, procedures, or plans to meet needs (e.g. a procedure from one setting transferred as is into another). The system will be employed within established operational limits. *Example:* The use of existing safety harnesses, in a conservative setting, to prevent aircrew injury while handling specialized cargo. *Occurrences*: 2	*Use existing*: Down-selecting for an existing subpopulation to meet needs. The users already possess the required skills, traits, and capacities needed without restrictive selection criteria that substantially reduce the population of possible users (e.g. only left-handed people who are taller than 2 meters). *Example:* The use of existing MC-130E aircrew to fly the mixed formation mission. *Occurrences:* 3
2	*Minor modification of existing:* The minor modification of existing material solutions using common technology and skills. As a heuristic, the baseline system is easily recognizable in spite of the change. Common technology is diffuse and not solely available in laboratory or industrial settings. These changes may be reversible (e.g. plug in) or may be permanent (e.g. drill several holes). *Examples:* Securing a commercial hunting scope to a military rifle with electrical tape. Grinding down the edge of a machete to have a shorter blade with a sharper edge. *Occurrences:* 9	*Configuration level change:* The exploration and employment of a novel system configuration; configurations are unique system states afforded by different settings and modules a system has (e.g. jeep (system) with top (module) and doors (modules) removed, headlights on (setting) and radio on (setting)). The modification of an existing procedure through the inclusion of new configurations, addition of existing configurations or through the alteration of timing (e.g. transition from one state to another faster or slower, reordering of sequence). *Examples:* Introducing a dissimilar throttle setting for the engines to alter aircraft stall and handling characteristics. Operating the ground mapping radar at its	*Task creation, improved efficiency and reliability:* The modification or creation of a new task for system users. Task accomplishment requires some combination of user attributes (e.g. knowledge, strength) and consumes resources (e.g. cognitive bandwidth). Tasks may be developed through a variety of means (e.g. experimentation, use, and training). Tasks that are already automated within a system may also be augmented by or transferred to system users. Task efficiency and reliability may be improved through repetition and training. *Examples:* The soldiers had to learn how to aim with both eyes open to use the commercial hunting scope. The aircrew had to learn how to fuse, to arm, and to safely handle the napalm

(continued)

Table 43.4 (continued)

Level	Material	Operational	Human
		extreme deflection due to the nose high attitude of the aircraft at low speed. *Occurrences:* 8	bomb/navigation marker. The aircrew modified how they processed their terrain avoidance radar display due to operation beyond the limits of the radar. *Occurrences:* 23
3	*Substantial modification or creation:* A modification of an existing material solution that may require specialized technology or skills. The development of novel component level material solutions. A substantial modification of an existing system is one where there is still more of the original system than the changes, yet it is not a minor modification. *Example:* Building a wooden cradle to serve as an interface between a napalm bomb and a drag chute; the cradle also permitted handling of the napalm bomb within the aircraft cargo bay. *Occurrences:* 2	*Procedure level change:* The creation of a new sequence of system configurations to accomplish discrete system functions. The extensive change to existing procedures such that an effectively new procedure is created. Alterations to an existing plan. *Examples:* Creating the procedures to handle and release a modified napalm bomb/navigation marker from the ramp of the aircraft to include determining the appropriate timing for accurate emplacement. Extensively altering the engine failure checklist to accommodate the change in starting state due to the dissimilar engine configuration. *Occurrences:* 5	*Specialization and system capacity increase:* The selection of specialized populations of personnel to introduce rare combinations of traits (e.g. cognitive, strength, reliable performance) to a system. The increase of system capacity through the addition of personnel. *Examples:* The addition of a navigator due to the increased workload from the newly added FLIR. The selection of extraordinary helicopter pilots for the cognitively intensive close formation flight with fixed wing aircraft. The intensive selection of ground forces for specific traits and demonstrated performance (~5% selected from an already competitive pool of >300). *Occurrences:* 8
4	*Creation of a new system:* Either the creation of a completely new system or the completion of extensive modifications to an existing system. As a heuristic, modifications are considered extensive if following additions, removals, and modifications, less than half of the resulting system is original.	*Plan level change:* The sequencing and integration of multiple procedures to realize system requirements. The extensive change to an existing plan such that an effective new plan is created. *Example:* The sequencing of multiple procedures to enable precise night time navigation	*Task structure creation*: The development of a complex task structure to assure system performance. The creation of complex control sequences, which require cumulative training and repetition to assure adequate and reliable performance. Some form of partial task training is

(continued)

Table 43.4 (continued)

Level	Material	Operational	Human
	Example (notional): The creation of a small and lightweight precision navigation system (e.g. suitable for helicopter installation). *Occurrences:* 0	of the two formations of mission aircraft, including procedures for formation loss/rejoin, formation maintenance, and reconfiguration due to lead aircraft failure. *Occurrences:* 3	anticipated. *Example:* The incremental build-up of formation flight between the helicopters and cargo aircraft required learning first how to fly together in the day time at constant altitude with communication and progressed in difficulty to night time, low-level terrain following flight with radio silence. *Occurrences:* 6

multicomponent system) material changes were assessed (thus only three levels are shown); Level 4 changes were retained due to the complex operational and human changes (e.g. Level 4), which were observed. The descriptions of each category, along with the number of changes observed, can be seen in Table 43.4.

43.4 Findings and Discussion

We will use this section to illustrate how the users employed changes to realize novel function. This section has two parts: first, a walkthrough of user change behaviors using illustrative examples and second, a discussion of these observations. Through our examples, we will show that the system users are relying upon changing how they use existing systems (operational change) more than changing the systems. There is theoretical support for operational change in the literature on reconfiguration [41, 42]; however, this research demonstrates how users can exploit this system aspect. Next, we will show that users are relatively unconstrained in their application of human changes, selecting people to enhance performance and transferring substantial system function to people. Finally, we will tie the changes we have observed to our motivating problem and the literature on flexible design.

43.4.1 Illustrative Examples

The low-speed MC-130E case illustrates how the users realized novel capability with an existing complex system; the changes employed were almost exclusively operational and human. The need for a high-precision, low-speed pathfinder arose from choices made early in mission design. To reduce the threat posed by the enemy, Operation Kingpin's leveraged the shock and surprise of a controlled crash of a heliborne assault team within the center of the prison camp. Among other capabilities, this required the precision landing of a small helicopter and its precision navigation to reach a point deep within mountainous enemy territory. At the time, these capabilities were individually possible on existing aircraft; however, no single aircraft had the required combination of capabilities.

Two helicopters were candidates for the mission, both the UH-1H and the HH-3E could carry an adequate assault team and both were small enough to land (albeit with significant rotor damage) within the prison yard. The candidate helicopters were capable of precision landing; however they possessed rudimentary navigation capabilities. As early as 1960, a decade before Operation Kingpin, fighter aircraft were integrating inertial navigation systems to realize precision navigation. When installed, these systems weighed just less than 80 lb [43]. Of the two helicopters, the UH-1H had the least capacity, yet it still had a payload capacity of roughly 2100 lb. when fully fueled and manned. In considering weight alone, integration of a contemporary precision navigation system was a feasible option.

Rather than integrating an existing precision navigation system on the helicopter (material change), the users pursued a solution that, at the offset, relied on operational change. The mission planners were aware of two existing capabilities relevant to navigation and helicopter operations. First, they knew that the MC-130E had precision navigation capabilities and was actively in use for terrain following missions within North Vietnam [35]. The planners knew that there were performance incompatibilities between the MC-130E and the candidate helicopters; however, they were aware of tactics recently pioneered for helicopter aerial refueling [44]. Air Force pilots, 5 years earlier, had discovered that helicopters could attain higher speeds and remain in formation with fixed wing aircraft if they were positioned on the fixed wing aircraft's wing tip vortex [45, 46]. Once discovered, this aerial surfing phenomenon was exploited for both helicopter aerial refueling and long-distance ferry missions [45]. Thus, instead of modifying the helicopters to have precision navigation, the users chose to guide the helicopters with an MC-130E and overcome the airspeed incompatibilities through using procedures pioneered for aerial refueling.

In going another level down, and following the integration efforts, we are able to see human changes. Aerial surfing was traditionally employed in level flight, yet the users wanted to employ it in a more dynamic low-level profile to avoid enemy radar detection. After initial experiments to configure the MC-130E to fly slow enough to join with the helicopters (altering flap and power settings), the users found that the aircraft was not responsive enough to perform in a dynamic profile. To regain control response, the users experimented with engine settings and redistributed power toward the inboard engines of the four-engine MC-130E. At the surface, we can see the operational changes and the materials employed; however, on further consideration, we see that each configuration change altered the behavior of the aircraft (e.g. slow speed resulting in sluggish response) and in turn required new skills and control behaviors for the pilots (human change). The transition from a uniform to a variable engine setting required a completely new control behavior and workload for the users. Also, the low speeds were beyond the operating capabilities of automated systems (e.g. autopilot) and thus drove an allocation of tasks from the system to the users.

The dominant changes in this setting were human changes. Although most of these changes were gains of simple skills (Level 2 in Table 43.4), the users also employed higher level changes (Levels 3 and 4). First, there were instances where the users chose extraordinary people who could make the system work as opposed to altering the material system (Level 3). Second, the users developed human task structures that required substantial investments in training to assure overall system performance (Level 4). The users altered their actions, skills, and capacities to implement operational changes and thus realize novel system behaviors; they underwent intensive training to alter themselves.

Although selection and task structure development are present in the low-speed MC-130E case, the clearer examples come from the night sight case. The users considered an already competitive population of Special Forces personnel and selected less than 5% of them to land directly within the prison. Their selection criteria were based on existing skills, traits, and demonstrated performance. Next, the users allocated numerous tasks to themselves, tasks that required complex structures once integrated. To integrate tasks and ensure reliable performance, the users engaged in intensive repetition and incremental build-up (slow to fast, day to night, isolated to integrated tasks). Even after they started with an incredibly skilled force, the users still needed to practice the mission six times a day for a month.

The users were relatively unconstrained in their use of human change when compared with traditional designers. First, in their use of selection, they chose people who could make the system work. In contrast, designers typically design for the largest population possible [47, 48]. Next, the users allocated numerous functions to themselves, a reverse of labor-saving trends. Through embodying system capability in themselves, and not the material system, users made a system with an incredibly high barrier to transfer. They created a system that required a substantial investment in time to attain desired performance. Designers typically seek low workloads and low barriers to transfer in their designs [47, 48].

43.4.2 Discussion of Results

The changes we observed were not permanent solutions, rather they were transient. We were able to observe how existing technologies can be rapidly changed to meet a one-time need. Although only short term, these changes have implications that tie back directly to our motivation to maintain functional relevance over a system's life cycle. First, changing requirements are not solely limited to long-term or population-wide shifts, one time or single user requirements can exist. The observed user changes permitted short-term, and for the most part reversible, change with complex systems to meet unforeseen needs. Second, through the benefit of the passage of time, we know that user innovations with broader utility were ultimately adopted, matured, and diffused. Thus, although the user changes were transient and focused on meeting short-term needs, they permitted a phase of exploration and variation that fed long-term system change and functional relevance. As an example, a first generation infrared sensor was crudely installed to support the mission (requiring the addition of a new crewmember to control it and integrate its data); although a temporary change, its utility was highlighted in the After Action Report and it was ultimately included in the next block modification program for the MC-130E (with an integration that did not necessitate an additional crewmember).

While there is literature that explores the relationship between flexibility and reconfigurable systems [42], this present work builds on existing concepts, particularly with manned systems. The user's ability to control the system and realize unexplored configurations and configuration sequences yielded new performance. In this minimally automated setting, these operational changes relied on the user's ability to create new control structures through exploration of available interfaces and learning. In some instances, this process relied on the virtuous performance of extremely skilled users. The operational changes we observed showcase how reconfigurable systems can enable variable function; the underlying human changes reveal at least one means for realizing these operational changes.

A theme within the literature on flexibility is the trade between optimizing a design for known performance and overdesigning to meet unknown future requirements [1, 3]. The user pathways to change point to similar trades, specifically trades with user interfaces and the users themselves. We see a push to streamline user interfaces for known requirements, particularly with complex systems [49]; in contrast to this, we see how users can access available interfaces in unanticipated manners to realize novel performance. The flexibility afforded through available configurations and the necessary interfaces to attain them point a need for research into the trade between optimal interfaces and those interfaces that afford operational flexibility. Next, the reliance on human changes to attain novel and improved performance indicates that a skilled user base (not merely the minimum) can enable system flexibility. Thus, trades associated with an overdesign of the human element of complex systems (e.g. capacity, skills, training) may also provide a source of flexibility.

43.5 Conclusion and Future Research

This work was motivated by the general need to design complex systems that can meet changing requirements. The existing literature in this area has explored several trades pertaining to system design aspects. In separate empirical research, we were able to observe traits highlighted by the literature (e.g. flexible and robust design). However, through broader analysis, we observed changes by system users, an actor who is typically constrained. This present research has focused on the phenomenon of user innovation with complex systems. We have attempted to unpack how users are able to realize change; hopefully, we have contributed useful insight into mechanisms to further realize flexible designs.

The fields of engineering and design recognize that there are often many ways to achieve desired functionality. The apocryphal space pen story is just one example of different paths to a common function: one path where the United States developed a million dollar pen and the other where the Russians merely used a pencil. Along these same lines, we have observed that users rely upon novel changes relative to the material changes supported by the flexibility literature. Rather than relying on complex or permanent material changes, the users exploited limited material change, complex operational changes, and an almost unconstrained change to themselves.

User pathways appear to be a stepping stone to more permanent changes, allowing for exploration and variation. Having these pathways is not free; as with the existing theories on flexible design, we see similar trades between optimal design for the known and overdesign for the unknown. User exploitation of unexplored system configurations highlights trades regarding system interface optimization. Similarly, human changes were in part enabled by an overdesigned (highly capable) workforce. We do not argue that user changes are better or less complex, we merely present them as alternative pathways to flexibility that merit further exploration.

Disclaimer The views expressed in this academic research paper are those of the author and do not reflect the official policy or position of the U.S. government or the Department of Defense.

References

1. De Neufville R et al (2004) Engineering systems monograph. In: Engineering systems symposium. Citeseer
2. Fricke E, Schulz AP (2005) Design for changeability (DfC): principles to enable changes in systems throughout their entire lifecycle. Syst Eng 8
3. Saleh JH, Mark G, Jordan NC (2009) Flexibility: a multi-disciplinary literature review and a research agenda for designing flexible engineering systems. J Eng Des 20:307–323
4. Cox A, Szajnfarber Z (2016) Post-production change to complex systems
5. Pyles RA (2007) Aging aircraft: USAF workload and material consumption lifecycle patterns (Santa Monica: RAND, 2003), xix
6. Hill OJ (2006) Aircraft modifications: assessing the current state of air force aircraft modifications and the implications for future military capability
7. Kendall F, Gilmore J, Halvorsen T (2015) Department of defense instruction 5000.02, operation of the defense acquisition system
8. Kendall F (2015) Implementation directive for better buying power 3.0 - achieving dominant capabilities through technical excellence and innovation
9. Cardin M-A (2014) Enabling flexibility in engineering systems: a taxonomy of procedures and a design framework. J Mech Des 136:11005

10. Ryan ET, Jacques DR, Colombi JM (2013) An ontological framework for clarifying flexibility-related terminology via literature survey. Syst Eng 16:99–110
11. Steiner R (1998) Systems architecture and evolvability-definitions and perspective. In: Proceedings of the 8th Annual Symposium of the International Council on System Engineering
12. De Neufville R, Scholtes S, Wang T (2006) Real options by spreadsheet: parking garage case example. J Infrastruct Syst 12:107–111
13. de Neufville R (2002) Architecting/designing engineering systems using real options
14. Von Hippel E (2007) The sources of innovation. Springer
15. Shah SK, Tripsas M (2007) The accidental entrepreneur: the emergent and collective process of user entrepreneurship. Strateg Entrep J 1:123–140
16. Shah SK, Mody C (2011) Innovation, social structure & the creation of new industries
17. Lüthje C, Herstatt C, Von Hippel E (2003) The dominant role of 'local' information in user innovation: the case of mountain biking
18. Von Hippel E (1994) 'Sticky information' and the locus of problem solving: implications for innovation. Manag Sci 40:429–439
19. Von Hippel E, Katz R (2002) Shifting innovation to users via toolkits. Manag Sci 48:821–833
20. Baldwin C, von Hippel E (2011) Modeling a paradigm shift: from producer innovation to user and open collaborative innovation. Organ Sci 22:1399–1417
21. von Hippel E (2005) Democratizing innovation. MIT Press, Cambridge
22. Riggs W, Von Hippel E (1994) Incentives to innovate and the sources of innovation: the case of scientific instruments. Res Policy 23:459–469
23. Dorst K, Cross N (2001) Creativity in the design process: co-evolution of problem–solution. Des Stud 22:425–437
24. Liikkanen LA, Perttula M (2009) Exploring problem decomposition in conceptual design among novice designers. Des Stud 30:38–59
25. Suwa M, Gero J, Purcell T (2000) Unexpected discoveries and S-invention of design requirements: important vehicles for a design process. Des Stud 21:539–567
26. Cross N (2004) Creative thinking by expert designers. J Des Res 4
27. Purcell AT, Gero JS (1996) Design and other types of fixation. Des Stud 17:363–383
28. Miles MB, Huberman AM (1994) Qualitative data analysis: an expanded sourcebook. Sage, Thousand Oaks
29. Eisenhardt KM (1989) Building theories from case study research. Acad Manag Rev 14:532–550
30. JCS Joint Contingency Task Group (1970) C. Report on the Son Tay Prisoner of War Operation
31. Gargus J (2010) The son tay raid: American POWs in Vietnam were not forgotten. Texas A&M University Press, College Station
32. Schemmer BF (2002) The raid: the son tay prison rescue mission. Ballantine Books, New York
33. Grimes B (2014) The history of big safari. Archway Publishing, Bloomington
34. McRaven WH, McRaven WH (1996) Spec ops: case studies in special operations warfare: theory and practice. Ballantine Books, Random House Publishing Group
35. Thigpen JL (2001) The praetorian STARShip: the untold story of the Combat Talon. Air University Press, Maxwell Air Force Base
36. Whitcomb DD (2012) On a steel horse I ride: a history of the MH-53 pave low helicopters in war and peace. DTIC Document, Maxwell Air Force Base
37. Yin RK (2013) Case study research: design and methods. Sage publications, London
38. Corbin J, Strauss A (2014) Basics of qualitative research: techniques and procedures for developing grounded theory. Sage publications, Los Angeles
39. Langley A (1999) Strategies for theorizing from process data. Acad Manag Rev 24:691–710
40. Cox A, Szajnfarber Z (2015) Case study research of user design methods. In: Proceedings of the International Annual Conference of the American Society for Engineering Management. 1
41. Ferguson S, Siddiqi A, Lewis K, de Weck OL (2007) Flexible and reconfigurable systems: Nomenclature and review. In: ASME 2007 International Design Engineering Technical

Conferences and Computers and Information in Engineering Conference. American Society of Mechanical Engineers, New York, pp 249–263
42. Siddiqi A, de Weck OL (2008) Modeling methods and conceptual design principles for reconfigurable systems. J Mech Des 130:101102
43. Lambert M (1963) F-104G Starfighter, European production of systems. Flight Int 2818:375
44. Lindsay J et al (1988) Son tay raid panel discussion
45. Hannan J (1993) Aerial Refueling: An Army Requirement for the AH-64 Apache? US Army Aviat Dig 2:32–39
46. Eastman W Development of helicopter air refueling. USAF helicopter pilot association, interesting experiences
47. Norman DA (2013) The design of everyday things: revised and expanded edition. Basic books, New York
48. Wickens CD (1992) Engineering psychology and human performance
49. Cummings M, Sasangohar F, Thornburg K, Xing J, D'Agostino A (2010) Human-system interface complexity and opacity part i: literature review. Massachusettes Inst Technol Camb MA

Part VI
Systems Science, Systems Thinking and Complexity Management

Chapter 44
Threshold Metric for Mapping Natural Language Relationships Among Objects

Joseph J. Simpson, Mary J. Simpson, and Thomas B. Kercheval

Abstract Formal system concepts, with direct connections to informal system representations, provide a "much-needed" pathway for the application of structure to a range of valuable systems methods, analyses, and techniques. The augmented model-exchange isomorphism (AMEI) provides a foundational link between formal and informal system representations. The AMEI focuses on the system structuring relationship that creates a system. This chapter expands that discussion to include the objects associated with a system and other contextual considerations.

Keywords Augmented model-exchange isomorphism • System structuring relationship • Structural integration modeling • Logical properties of reflexivity, symmetry and transitivity

44.1 Overview

A standard framework for structural modeling is required to effectively communicate system structure. This chapter seeks to establish a threshold number of objects needed, in combination with a given relation, to effectively establish a system. Using this metric, we may verify the mapping of a specific natural language relationship with a given number of objects contained in the system model under development. Establishing this threshold will provide a basis for determining which logical relation types, specified by the [1] augmented model-exchange isomorphism (AMEI), may be valid within a given set of objects. The threshold metric has two different uses depending on whether we are designing or discovering the system of interest.

Structural modeling [2], as developed by Warfield, is separated into two components: basic structural modeling (BSM) and interpretive structural modeling (ISM). The first component, BSM, is the realm of mathematicians. Contained within BSM are the abstractions of structure needed to encode relations between and among objects in a mathematically valid way. Verification and validation of

J.J. Simpson (✉) • M.J. Simpson • T.B. Kercheval
System Concepts, Seattle, WA, USA
e-mail: jjs0sbw@gmail.com; mjs0sbw@gmail.com

these models can be achieved using the laws of logic and mathematics, indicating that the encoded structure is properly aligned and configured. Due to the abstract nature of BSM, BSM can be applied to a wide variety of different domains.

The second component of Warfield's structural modeling, ISM, is the realm of domain experts. ISM was developed to organize empirical, substantive knowledge of complex real-world systems/issues. As a result, the structure of models developed by an ISM process may not be as well defined as their BSM counterparts. Several different models, or structures, may be produced by an ISM process, elaborating on different perspectives of the same system/issue.

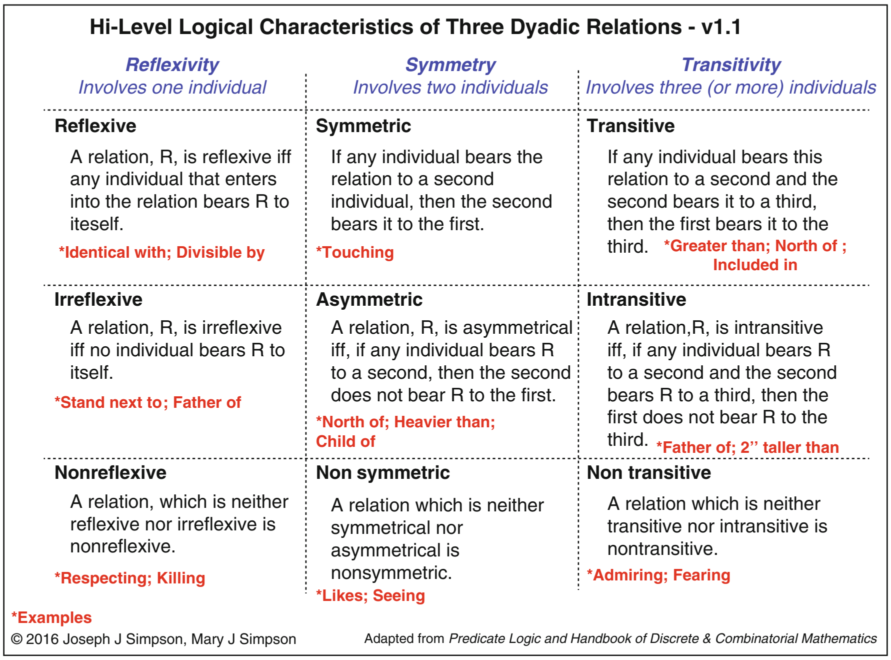
These two components of structural modeling are each powerful tools in their own right. However, when considered individually, each lacks the strengths of the other and both fall short of being a robust, general methodology for modeling complex and/or unknown systems. To account for and reduce the cognitive complexity intrinsic to modeling large-scale systems, BSM and ISM must be put in "proper correspondence" [3].

A third component, structural integration modeling (SIM) was introduced to properly align BSM and ISM by Simpson & Simpson [4]. SIM has many artifacts including the abstract relation types (ART) and the AMEI. ART [5] were introduced to effectively package and communicate information associated with a SIM activity. Coupling prose (formal/informal), structured graphs, and mathematics in the standardized ART form heighten the probability of successful information transfer. The AMEI creates a standard framework for mapping natural language relationships into mathematically valid relations. Every relation specified by the AMEI has a well-defined set of logical properties, enabling the application of processes, which are in proper correspondence with the relation of interest [1].

Well-defined relations are foundational to effectively documenting and communicating system structure and information. For the purposes of this chapter, a system is defined by the "construction-rule" definition [6], "A system is a relationship mapped over a set of objects." Uncertainty concerning the logical properties of the system structuring relation is a major source of cognitive complexity associated with the system as a whole. Use of the AMEI facilitates the positioning of the natural language system structuring relationship in proper correspondence with the mathematics used to encode the structure of the system of interest, thereby reducing the uncertainty and complexity associated with a modeling activity and the subject system structure.

44.2 Natural Language Relationships and Logical Properties

Since the AMEI is concerned with the natural language relationship mapped between and among objects, systems that can be considered within its scope must contain more than one object. While it may be possible to have a system comprised



Hi-Level Logical Characteristics of Three Dyadic Relations - v1.1

Reflexivity *Involves one individual*	***Symmetry*** *Involves two individuals*	***Transitivity*** *Involves three (or more) individuals*
Reflexive A relation, R, is reflexive iff any individual that enters into the relation bears R to iteself. ***Identical with; Divisible by**	**Symmetric** If any individual bears the relation to a second individual, then the second bears it to the first. ***Touching**	**Transitive** If any individual bears this relation to a second and the second bears it to a third, then the first bears it to the third. ***Greater than; North of ; Included in**
Irreflexive A relation, R, is irreflexive iff no individual bears R to itself. ***Stand next to; Father of**	**Asymmetric** A relation, R, is asymmetrical iff, if any individual bears R to a second, then the second does not bear R to the first. ***North of; Heavier than; Child of**	**Intransitive** A relation,R, is intransitive iff, if any individual bears R to a second and the second bears R to a third, then the first does not bear R to the third. ***Father of; 2” taller than**
Nonreflexive A relation, which is neither reflexive nor irreflexive is nonreflexive. ***Respecting; Killing**	**Non symmetric** A relation which is neither symmetrical nor asymmetrical is nonsymmetric. ***Likes; Seeing**	**Non transitive** A relation which is neither transitive nor intransitive is nontransitive. ***Admiring; Fearing**

***Examples**

© 2016 Joseph J Simpson, Mary J Simpson

Adapted from *Predicate Logic and Handbook of Discrete & Combinatorial Mathematics*

Fig. 44.1 Logical properties

of only one object, all of the structure associated with such a system would be completely unknown. To gather structural information about such a black-box system, one would have to decompose it into its internal objects and map relations between and among those. Once such a decomposition has been performed, the system model would now consist of multiple objects and be within the scope of the AMEI.

Within the AMEI, there are three groupings of logical properties: symmetry, reflexivity, and transitivity (see Fig. 44.1). Every relation given by the AMEI must have one member from each of these groupings.

Reflexivity involves one individual object. The logical properties constituent to the Reflexivity grouping are the reflexive, irreflexive, and nonreflexive property. If a relation is reflexive, then an object bears this relation to itself (*x*R*x*). An irreflexive relation states that no object bears this relation to itself (*x*’R*x*). The nonreflexive logical property is a composite property, which states that in a set of objects, some objects are reflexive and some objects are irreflexive.

Symmetry involves two individual objects. The symmetric, asymmetric, and nonsymmetric logical properties belong to the Symmetry grouping. A symmetric relation requires that if object *x* bears a relation to object *y*, then object *y* also bears a relation to object *x* ((if *x*R*y*, then *y*R*x*) and (*x* != *y*)). An asymmetric relation states that if object *x* bears a relation to object *y*, then object *y* does not bear a relation to object *x* ((if *x*R*y*, then *y* ’R*x*) and (*x* != *y*)). The nonsymmetric logical property is a composite property and can only exist when a set of objects have both symmetric and asymmetric relations mapped among them.

Transitivity involves three or more individual objects. Transitive, intransitive, and nontransitive relations all belong to the transitivity grouping. Transitive relations state that if object *x* bears a relation to object *y* and object *y* bears a relation to object *z*, then object *x* also bears a relation to object *z* ((if (*x*R*y* and *y*R*z*), then *x*R*z*) and (*x* != *y* != *z*)). Intransitive relations state that if object *x* bears a relation to object *y* and object *y* bears a relation to object *z*, then object *x* does not bear a relation to object *z* ((if (*x*R*y* and *y*R*z*), then *x*'R*z*) and (*x* != *y* != *z*)) The nontransitive logical property is a composite property and may only exist where a set of objects have both transitive and intransitive relations mapped among them.

These logical properties are used in a group that contains one logical property from each category. The logical property groups reduce complexity and uncertainty by clearly stating the logical property configuration associated with any given situation. These logical property groups are used in the SIM area of structural modeling and may not have valid, direct representations in either the BSM area or the ISM area. As the AMEI work is developed, the areas without direct representations will be identified and documented.

44.3 Approach for Analysis

The system design mode and discovery mode [6] are used to guide the analysis of the number of required objects in any system representation activity. As shown in Fig. 44.2, the design mode starts with a known system relationship, while the discovery mode starts with a set of known system objects. When the system concept is used in the design mode, the system relationship of interest is known, along with its associated logical property group.

When the system concept is used in the discovery mode, the system objects are known; the system relationship, along with its associated logical property group, must be determined.

When the mathematical aspects of BSM control the analysis, the *logical property that requires the highest number of objects determines the number of objects* needed in the system design mode. Reflexivity requires only one object. Symmetry requires two objects. Transitivity requires three objects. If a system structuring relationship is transitive, then the minimum number of objects in the system will be three (3). If the system structuring relationship is nontransitive, then the minimum number of objects is four (4). If the system structuring relationship is intransitive, then the minimum number of objects is three (3). These observations are reflected in Fig. 44.3, Permutations for Design Mode.

If the natural language logical properties from ISM control the analysis, then the minimum number of objects may change. Consider the "included-in" natural language relationship. This relationship is irreflexive, asymmetric, and transitive. It is possible to design a container that includes a liquid. This system has only two

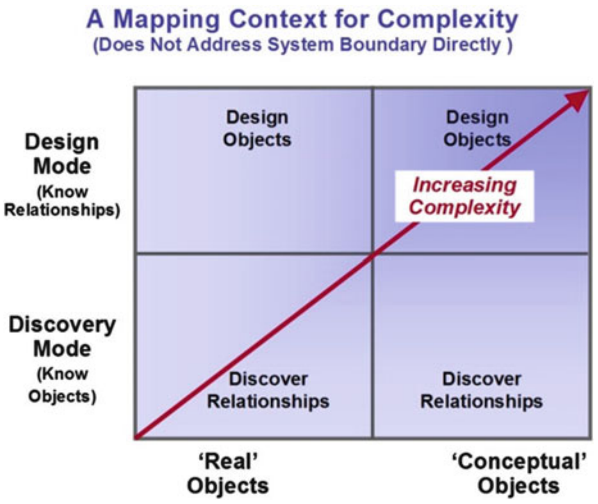

A Mapping Context for Complexity
(Does Not Address System Boundary Directly)
Design Mode (Know Relationships)
Discovery Mode (Know Objects)
Design Objects
Design Objects
Increasing Complexity
Discover Relationships
Discover Relationships
'Real' Objects
'Conceptual' Objects

Fig. 44.2 System modes

Permutations of Relation Properties with Unique Identifiers – v1.5 (for Design Mode)					
RST- [1,1,1] #Objects=**3**	**R**eflexive, **S**ymmetric, **T**ransitive	RSI- [1,1,2] #Objects=**3**	**R**eflexive, **S**ymmetric, **I**ntransitive	RSN- [1,1,3] #Objects=**4**	**R**eflexive, **S**ymmetric, **N**ontransitive
IST- [2,1,1] #Objects=**3**	**I**rreflexive, **S**ymmetric, **T**ransitive	ISI- [2,1,2] #Objects=**3**	**I**rreflexive, **S**ymmetric, **I**ntransitive	ISN- [2,1,3] #Objects=**4**	**I**rreflexive, **S**ymmetric, **N**ontransitive
NST- [3,1,1] #Objects=**3**	**N**onreflexive, **S**ymmetric, **T**ransitive	NSI- [3,1,2] #Objects=**3**	**N**onreflexive, **S**ymmetric, **I**ntransitive	NSN- [3,1,3] #Objects=**4**	**N**onreflexive, **S**ymmetric, **N**ontransitive
RAT- [1,2,1] #Objects=**3**	**R**eflexive, **A**symmetric, **T**ransitive	RAI- [1,2,2] #Objects=**3**	**R**eflexive, **A**symmetric, **I**ntransitive	RAN- [1,2,3] #Objects=**4**	**R**eflexive, **A**symmetric, **N**ontransitive
IAT- [2,2,1] #Objects=**3**	**I**rreflexive, **A**symmetric, **T**ransitive	IAI- [2,2,2] #Objects=**3**	**I**rreflexive, **A**symmetric, **I**ntransitive	IAN- [2,2,3] #Objects=**4**	**I**rreflexive, **A**symmetric, **N**ontransitive
NAT- [3,2,1] #Objects=**3**	**N**onreflexive, **A**symmetric, **T**ransitive	NAI- [3,2,2] #Objects=**3**	**N**onreflexive, **A**symmetric, **I**ntransitive	NAN- [3,2,3] #Objects=**4**	**N**onreflexive, **A**symmetric, **N**ontransitive
RNT- [1,3,1] #Objects=**3**	**R**eflexive, **N**onsymmetric, **T**ransitive	RNI- [1,3,2] #Objects=**3**	**R**eflexive, **N**onsymmetric, **I**ntransitive	RNN- [1,3,3] #Objects=**4**	**R**eflexive, **N**onsymmetric, **N**ontransitive
INT- [2,3,1] #Objects=**3**	**I**rreflexive, **N**onsymmetric, **T**ransitive	INI- [2,3,2] #Objects=**3**	**I**rreflexive, **N**onsymmetric, **I**ntransitive	INN- [2,3,3] #Objects=**4**	**I**rreflexive, **N**onsymmetric, **N**ontransitive
NNT- [3,3,1] #Objects=**3**	**N**onreflexive, **N**onsymmetric, **T**ransitive	NNI- [3,3,2] #Objects=**3**	**N**onreflexive, **N**onsymmetric, **I**ntransitive	NNN- [3,3,3] #Objects=**4**	**N**onreflexive, **N**onsymmetric, **N**ontransitive

Fig. 44.3 Permutations for design mode

objects, the container and the liquid. A third object is not needed to support the transitive nature of this natural language relationship. The apparent conflict represented by differing numbers of required objects is resolved in SIM. Clear definitions must be created for each specific operational instance. Figure 44.3 reflects the number of objects required with BSM controlling.

Permutations of Relation Properties with Unique Identifiers – v1.5 (for Discovery Mode)					
RST- [1,1,1] #Objects=**4**	**R**eflexive, **S**ymmetric, **T**ransitive	RSI- [1,1,2] #Objects=**4**	**R**eflexive, **S**ymmetric, **I**ntransitive	RSN- [1,1,3] #Objects=**4**	**R**eflexive, **S**ymmetric, **N**ontransitive
IST- [2,1,1] #Objects=**4**	**I**rreflexive, **S**ymmetric, **T**ransitive	ISI- [2,1,2] #Objects=**4**	**I**rreflexive, **S**ymmetric, **I**ntransitive	ISN- [2,1,3] #Objects=**4**	**I**rreflexive, **S**ymmetric, **N**ontransitive
NST- [3,1,1] #Objects=**4**	**N**onreflexive, **S**ymmetric, **T**ransitive	NSI- [3,1,2] #Objects=**4**	**N**onreflexive, **S**ymmetric, **I**ntransitive	NSN- [3,1,3] #Objects=**4**	**N**onreflexive, **S**ymmetric, **N**ontransitive
RAT- [1,2,1] #Objects=**4**	**R**eflexive, **A**symmetric, **T**ransitive	RAI- [1,2,2] #Objects=**4**	**R**eflexive, **A**symmetric, **I**ntransitive	RAN- [1,2,3] #Objects=**4**	**R**eflexive, **A**symmetric, **N**ontransitive
IAT- [2,2,1] #Objects=**4**	**I**rreflexive, **A**symmetric, **T**ransitive	IAI- [2,2,2] #Objects=**4**	**I**rreflexive, **A**symmetric, **I**ntransitive	IAN- [2,2,3] #Objects=**4**	**I**rreflexive, **A**symmetric, **N**ontransitive
NAT- [3,2,1] #Objects=**4**	**N**onreflexive, **A**symmetric, **T**ransitive	NAI- [3,2,2] #Objects=**4**	**N**onreflexive, **A**symmetric, **I**ntransitive	NAN- [3,2,3] #Objects=**4**	**N**onreflexive, **A**symmetric, **N**ontransitive
RNT- [1,3,1] #Objects=**4**	**R**eflexive, **N**onsymmetric, **T**ransitive	RNI- [1,3,2] #Objects=**4**	**R**eflexive, **N**onsymmetric, **I**ntransitive	RNN- [1,3,3] #Objects=**4**	**R**eflexive, **N**onsymmetric, **N**ontransitive
INT- [2,3,1] #Objects=**4**	**I**rreflexive, **N**onsymmetric, **T**ransitive	INI- [2,3,2] #Objects=**4**	**I**rreflexive, **N**onsymmetric, **I**ntransitive	INN- [2,3,3] #Objects=**4**	**I**rreflexive, **N**onsymmetric, **N**ontransitive
NNT- [3,3,1] #Objects=**4**	**N**onreflexive, **N**onsymmetric, **T**ransitive	NNI- [3,3,2] #Objects=**4**	**N**onreflexive, **N**onsymmetric, **I**ntransitive	NNN- [3,3,3] #Objects=**4**	**N**onreflexive, **N**onsymmetric, **N**ontransitive

Fig. 44.4 Permutations for discovery mode

The logical property that requires the highest number of objects also determines the number of objects needed in the system discovery mode. However, in the discovery mode, the logical property set may change beyond the minimum set as more objects are discovered. The uncertainty associated with the logical property group in the discovery mode is a natural consequence of the discovery activity. New information can be discovered at any time, which has the capability of changing the current logical property group elements. Therefore, in the discovery mode, the transitive logical property needs four (4) objects, the intransitive logical property needs four (4) objects, and the nontransitive property needs four (4) objects, as shown in Fig. 44.4 – Permutations for Discovery Mode.

The minimum number of objects in the design mode may also be questioned on the basis of the known relationship properties. If a nontransitive system, in design mode, has only three objects, then one or more other objects must be added to complete the logic of the system's structure. The nontransitive system with three objects may be called incomplete, but this is a semantic detail that should be directly addressed in the context of any given system design activity. This chapter uses the minimum number of objects given in Fig. 44.3; incomplete systems are not considered.

As shown in this analysis, the system mode (design or discovery) will impact the minimum number of objects required to represent or determine the proper set of logical properties. Other contextual conditions may also impact the minimum number of system objects.

44.4 Contextual Considerations

The minimum number of objects listed in Figs. 44.3 and 44.4 are all associated *with a single thread of system structure*. In many system discovery activities, there may be more than one thread of system structure. The DOCLUS command and an example from Warfield's work [7] are used here to support a more detailed discussion of these issues. In the referenced example, there are thirteen (13) objects that need to be clustered into categories. The natural language system structuring relationship for this example is "in the same category as" (ITSCA), which has the following logical property set: reflexive, symmetric, and transitive.

There are four categories in this example and one system structural thread for each category. As a result, it would be possible, during the system discovery phase, to evaluate four objects and find that they had no relationship to each other. However, the fifth object would have to be associated with one of the first four objects. In a similar manner, twelve (12) of the objects could be in three groups, leaving one object only for the fourth group.

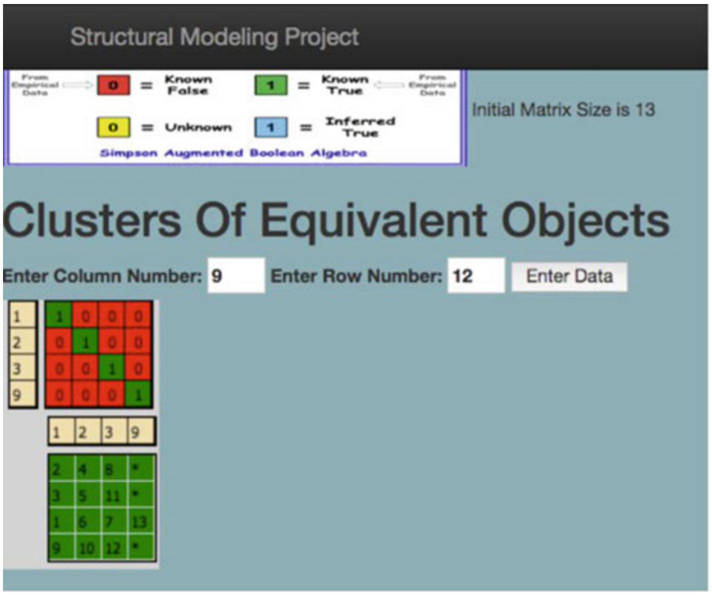
These contextual considerations inject more information and additional logical considerations to assist in the correct system structuring. Given these types of situations, the minimum number of objects in a system should only be applied to single system structural threads. An executable example of the selected structuring problem is available as part of the Structural Modeling Project General Structural Model (SMP GSM) application, which can be downloaded from Github at: https://github.com/jjs0sbw/n1.

Each specific case of structural modeling will have contextual considerations that may impact the system structuring process. However, the guidelines presented here for the minimum number of objects associated with each specific logical property group will help guide further exploration, development, and structured communication of system information.

In the referenced structuring activity [7] that clusters objects into similar categories, the final outcome is four (4) distinct clusters of objects. Figure 44.5 depicts the final system configuration. The four (4) clusters are numbered 1, 2, 3, and 9. It is interesting to notice that the only green cell backgrounds are on the diagonal in the top matrix. All other cells are marked with a red background, which means that these are not in the same category. The four structural threads in this example contain the following objects:

- Cluster One: 1, 6, 7, 13
- Cluster Two: 2, 4, 8
- Cluster Three: 3, 5, 11
- Cluster Nine: 9, 10, 12

Another Warfield example [7] orders a group of items based on their weight. This example, from Section A2.2.1, orders a set of seven objects. The seven objects being ordered are (1) feather, (2) Mack Truck, (3) beer can, (4) Volkswagen,


Structural Modeling Project
From Empirical Data
0 = Known False
1 = Known True
From Empirical Data
0 = Unknown
1 = Inferred True
Simpson Augmented Boolean Algebra
Initial Matrix Size is 13
Clusters Of Equivalent Objects
Enter Data

Enter Column Number: 9 Enter Row Number: 12

	1	2	3	9
1	1	0	0	0
2	0	1	0	0
3	0	0	1	0
9	0	0	0	1

1	2	3	9
2	4	8	*
3	5	11	*
1	6	7	13
9	10	12	*

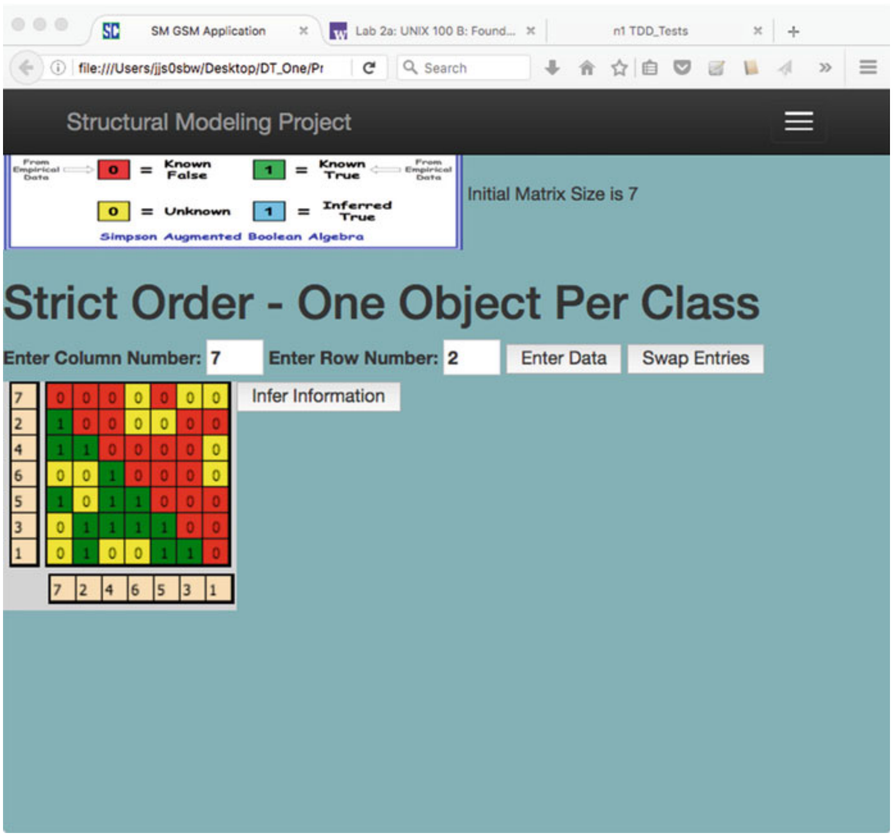
Fig. 44.5 Final configuration for “ITSCA” example

(5) small boy, (6) professional wrestler, and (7) universe. The logical property group associated with the natural language structuring relationship “is heavier than” is irreflexive, asymmetric, and transitive. In this case, “is heavier than” is also global and does not depend on the local interaction between two objects [8]. These are important contextual and semantic considerations that need to be incorporated with the findings from the AMEI and the minimum number of objects considerations. Figure 44.6 depicts the final system configuration for the “is heavier than” example.

Natural language system structuring relationships that are global generate a single structural thread through the structural graph. Global relationships that are asymmetric are acyclic [9] and do not generate cycles in the structural graphs. Natural language relationships that are local may generate multiple threads through the structural graph and generate a wide range of cycles as well.

44.5 Summary and Conclusions

The primary purpose of structural modeling is the discovery, identification, and communication of unknown system structural types. The most commonly evaluated areas are associated with groups of problems that are barriers to progress. Structural modeling is divided into three basic components: BSM, ISM, and SIM. The formal language and semantics of mathematics are the controlling language in the BSM



7	0	0	0	0	0	0	0
2	1	0	0	0	0	0	0
4	1	1	0	0	0	0	0
6	0	0	1	0	0	0	0
5	1	0	1	1	0	0	0
3	0	1	1	1	1	0	0
1	0	1	0	0	1	1	0
	7	2	4	6	5	3	1

Fig. 44.6 Final configuration for "is heavier than" example

area. Informal natural language is the controlling language in the ISM area. A mix of formal and informal languages is used in the SIM area to properly align the BSM and ISM structural modeling components.

The AMEI was developed as a structural modeling aid in the SIM area. The AMEI aligns prose and mathematics associated with natural language system structuring relationships in a manner that establishes a well-known standard. A system is composed of a relationship mapped over a set of objects. In this chapter, the minimum number of objects required to create a system was evaluated and presented based on two different system concept modes: discovery mode and design mode.

The general idea of global relationships and local relationships was also introduced to support the discussion of system structural threads that may be either cyclic or acyclic. These are important contextual considerations that are not directly aligned with the logical properties or minimum number of system objects. Taken together, these system metrics provide a robust set of system structuring guidelines that are useful in the discovery and communication of unknown system structure.

References

1. Simpson JJ (2015) Augmented Model-Exchange Isomorphism Version 1.1. doi:10.13140/2.1.3948.6241
2. Warfield JN (1994) A science of generic design, managing complexity through system design, 2nd edn. Iowa State University Press, Iowa
3. Warfield JN (1974) Structuring complex systems. Monograph no. 4. Battelle Memorial Institute, Ohio
4. Simpson JJ, Simpson MJ (2015) Structural modeling framework. Proceedings of the 25th Annual INCOSE international symposium. INCOSE, Seattle
5. Simpson JJ, Simpson MJ (2015) Objects, relations and clusters for system analysis. Proceedings of the 25th Annual INCOSE international symposium. INCOSE, Seattle
6. Simpson JJ, Simpson MJ (2006) Formal system concepts. Proceedings of the 4th Annual conference on systems engineering research. CSER, California
7. Warfield JN, Cardenas AC (1994) A handbook of interactive management, 2nd edn. Iowa State University Press, Iowa
8. Simpson JJ, Simpson MJ (2014) System analysis and identification: objects, relations and clusters. doi:10.13140/2.1.2821.1201
9. Warfield JN (1973) Binary matrices in system modeling. IEEE Trans Syst Man Cybern SMC-3 (5):441–449

Chapter 45
On the Nature of Systems Thinking and Systems Science: Similarities, Differences, and Potential Synergies

Len Troncale

Abstract This chapter clearly discriminates between the too-often-conflated terms – systems thinking (ST) and systems science (SS). It argues for a halt to this conflation. It presents a more comprehensive list than usual of different systems sources that must be unified to enable SE's to learn and use SS to increase use of its vast unused literature in Systems Engineering (SE). It presents a linear spectrum rather than opposition approach with thinking, philosophy, and design near one end and several new systems-integrated sciences (including SS) at the other end. It suggests several differences and similarities between thinking and science so similarities could enable synergy and complementary improvement within and between the two mega-domains. Balancing the current emphasis on human-based ST with increased study and use of the new SS would result in a more rigorous, prescriptive, and evidence-based Systems Engineering of greatly increased application range.

Keywords Systems thinking • Systems science • Isomorphic systems processes • Systems process engineering • Systems processes theory • Systems mega-domains

45.1 Statement of the Problem: Need to Address Directly

Systems thinking (ST) is a phrase often used in the systems engineering field. A whole range of SE meetings (IW, IS, CSER, Chapters) use the term "systems thinking" exclusively to represent SE interests in the systems approach often using the phrase equally and interchangeably with systems science (SS). The work and workers cited for this ST are clearly useful for some aspects of SE praxis. But the available knowledge important to near-future SE praxis is much broader than that implied by the term "systems thinking." SE must recognize that there are projects, problems, funding, and a vast, untouched literature that represents an

L. Troncale (✉)
Cal Poly University, 3801 W. Temple, Pomona, CA 91768, USA
e-mail: lrtroncale@cpp.edu

A.M. Madni et al. (eds.), *Disciplinary Convergence in Systems Engineering Research*, https://doi.org/10.1007/978-3-319-62217-0_45

emerging science of systems. It would foretell and enable a much broader view of SE that includes husbanding and repair of a wide range of natural systems, human systems, and complex combinations or hybrids of natural and human systems. This expanded SE scope will require a deeper view of how systems work and do not work (a much deeper systems theory) than simply project management. It will require a developed "science" of systems. But until systems engineers and systems scientists unify the currently very fragmented source information, there will not be a "SS" despite the loose use of such a term by many workers. Two ongoing projects of the INCOSE-SSWG (International Council on Systems Engineering - Systems Science Working Group) are trying to develop a framework to help synthesize, unify, or integrate the very fragmented areas of systems approach. One purpose of this chapter is to directly counter the conflation of the terms ST and SS. They are not and never will be the same.

45.1.1 Background of the Problem: A Long Tradition of Separation

Any conflation between the terms "ST" and "SS" is partly a result of already separated disciplines, the oft-cited "stovepipe" metaphor. Not only are disciplines made by beginning with distinct goals, often studying different scales of reality along the otherwise unbroken sequence of origins, but they use different tools, methods, and have different guarantors of truth. To a group of systems-interested advocates such as this audience, we do not need to emphasize that the recent trend is to decry and bemoan the separation of fields at a time when we need whole systems solutions to crisis problems. When we began our professional career in 1970, interdisciplinary was a scorned word. Now it is a necessity.

But even today, when even transdisciplinary is desired, remnants of the "science wars" continue [1, 2]. In the nineties, there was a public outbreak of articles in major journals representing both sides. Postmodernists stated logical positivism and reductionism were dead and did not produce factual knowledge, but rather knowledge was socially constructed and so inherently subjective and human-based. Scientists stated that the postmodernists had rejected objectivity as a possibility, objective methods in science, realism, and therefore much of the scientific knowledge base (KB) even though they were using its technological products daily. Scientists considered fields such as cultural anthropology, literature, history, and philosophy of science as hotbeds of antiscience rhetoric and teaching. They argued that most of these critics of science had a very poor knowledge of science to begin with. This was not simply separate foundational disciplines keeping knowledge separated, it was aggression against scientific knowledge and values. Nothing could be more inhibitory to the emergence of a science of systems than this. In my talks and numerous invited webinars to INCOSE engineers [3], I have encountered representatives of both of these camps.

45.1.2 Negative Effects of this Tradition

One could elect to view this traditional separation, antagonism, and mutual interference with understanding of the natural human need to preserve one's own culture due to fear of threat or to preserve past investment. However, there are several real and harmful outcomes to this human tendency, namely: (i) it places a great burden on the communications and cross-comparisons that are the ultimate source of desirable cross-fertilization; (ii) it inhibits needed attempts at unification, synthesis, and integration; (iii) it constrains indispensable awareness of the other areas and their significance; (iv) it halts utilization of others KBs; (v) it precludes teaching and learning across the areas by both the previous generation of professionals and the upcoming new generation of students; (vi) it widens the chasm developing between the approaches; and (vii) it lessens an appreciation of the value of approaches different from one's own.

45.1.3 Need for a Unified Systems KB and an Expanded View of SE

How can the new systems approach avoid the pitfalls of their foundational specialties that are just now becoming aware of the need for systems awareness yet simultaneously resisting each other? What would bring them together? Ironically, this dilemma is not that different from the one facing our international spectrum of national priorities or our political spectrum in the United States today. It is the goal of two of the several official projects of the INCOSE-SSWG – one on systems processes theory and second on systems pathologies (hereafter SPT/SP) – to discover strategies to overcome these barriers. First, we will attempt to provide a common terminology/ontology of such utility that it attracts proponents to replace opponents [11–15]. Second, we will use a framework of how systems work (SPT) and how they do not work (SP) [18] to produce an integrated KB of such detail it becomes a valued and widely taught and used tool in the SE toolbox. Third, we will produce an image of systems engineering that opens jobs and funding to SE for a much wider set of applications. Imagine an SE that effectively became the place to go to design not just aerospace or manufacturing products but to curate and repair a wide range of human/natural systems complexes on all scales. Fourth, that KB will be presented in products that could be easily used and adapted for educating a new generation of SEs. But can any of this be accomplished if there is hostility between human-based ST and natural science-based SS?

Table 45.1 Lifeworks in systems to be integrated: Output of ISSS workshop, July, 2013, San Jose, CA, plus authors personal additions shown in bold

1. **Abraham, R. (chaos math) [SS]**
2. Ackoff, R. (sys management) [ST]
3. **Allen, T. (hierarchies) [SS]**
4. Ashby, R. (sys management) [ST]
5. **Auyung, S. (complex systems) [SiS]**
6. Axelrod, R.M. (cooperation) [ST]
7. Bahm, A. (sys philosophy) [ST]
8. Banathy, B. (systems education) [ST]
9. **Bak, Per (self-criticality) [SS]**
10. Bar Yam, Y. (NECSI projects) **[SS]**
11. Barabasi, A.L. (network theory) [SS]
12. **Barrow, J. D. (physics) [SiS]**
13. Bateson, G. (sys philosophy) [ST]
14. Beer, Stafford (sys management) [ST]
15. Bertalanffy, Ludwig von (GST) **[SS]**
16. **Bogdanov, A. (organiz'l GST) [ST]**
17. **Bosch, Ockie (Bayesian apps)** [ST]
18. Boulding, K. (GST & econ) [SS]
19. Bunge, Mario (sys philosophy) [ST]
20. Cabrera, D. (systems dynamics) [ST]
21. Callon, (actor net theory) [ST]
22. Capra, F. (chronicler) [ST]
23. Caws, Peter (sys philosophy) [ST]
24. Checkland's, P. (soft systems) [ST]
25. Chomsky, N. (sys linguistics) [ST]
26. Churchman's, C. (sys mgmnt) [ST]
27. **Corning, Peter (synergy, bio) [SS]**
28. **Cowan's, G. (SFI) [SS]**
29. Doxiadis (ekistics) [ST]
30. **Eigen, Manfred (hypercycles) [SS]**
31. Foerster, H., von (cybernetics) [ST]
32. Forrester, Jay (syst dynamics) [ST]
33. **Francois, Charles (encyclopedia) [ST]**
34. Fuller, B. (sys architectures) [ST]
35. **Garajidajeh, J. (sysmanagement) [ST]**
36. **Gel Mann, Murray (flexions) [SS]**
37. **Gerard, Ralph (systems neurosci) [ST]**
38. **Haken, Herbert (synergy, phys) [SS]**
39. **Hall, A.D. (metasystems) [ST]**
40. **Hammond, D. (sys history) [ST]**
41. Holland's (agent-based modeling) [SS]
42. **Hood, Lee, systems biology [SiS]**

(continued)

Table 45.1 (continued)

43. **Iberall, A.S. (viable systems) [SS]**
44. **Jackson, Michael (CriticalSM) [ST]**
45. Jantsch, E. (sys philosophy) [ST]
46. **Karplus, M. (syschemistry) [SiS]**
47. Kauffman, Stuart (emergence) [SS]
48. Klir, G. (sys math, fuzzy sets) [SS]
49. **Langton's (artificial life) [SiS]**
50. Latour (actor net theory) [SS]
51. Laszlo, I. (systems philosophy) [ST]
52. **Leontief, W. (sys economics) [SS]**
53. **Lin, (gen'l systems theory) [SS]**
54. **Lorenz, Konrad (chaos) [SS]**
55. Mandelbrot, B. (fractals) [SS]
56. Maturana, H. (autopoiesis) [ST]
57. Mead, M. (sys anthropology) [ST]
58. Meadows, Donella (sys dynamics) [ST]
59. **Mesarovic, Mihalo (sys biology) [SiS]**
60. **Midgley, Gerald (interventions) [ST]**
61. Miller, James (living systems) [SS]
62. Mitchell, Melanie (complex sys) [SS]
63. Mitroff, Ian (sys management) [ST]
64. Odum, Howard (systems ecology) [SS]
65. **Pattee, H. (hierarch theory) [SS]**
66. Prigogine, Ilya (thermodynamics) [SS]
67. Rapoport, Anatol (game theory) [SS]
68. **Randall, Lisa (physics) [SiS]**
69. Richmond, B. (sysdynamics) [ST]
70. Ring, Jack (sys engineering) [ST]
71**. Rousseau, D. (sys philosophy) [ST]**
72. **Salthe, Stan (hierarchies) [SS]**
73. Senge, P. (systems management) [ST]
74. Shannon's (information theory) [SS]
75. Simms, Jim (quant living sys) [SS]
76. Simons, H.A. (computer systems) [SS]
77. **Skyttner, Lars (chronicler) [ST]**
78. Starkerman, (sys dynamics) [ST]
79. Sterman, (sys dyamics) [ST]
80. Strogatz, S. (chronicler) [ST]
81. Thom, R. (catastrophe theory) [SS]
82. Troncale, Len (sysprocess theory) [SS]
83. Varella, Francisco (autopoiesis) [ST]
84. Vesterby, V. (gen'l sys theory) [ST]
85. Vickers, Geoffrey (sysmanagement) [ST]

(continued)

Table 45.1 (continued)

86. Warfield's, John (ISM-managemnt) [ST]
87. **Weinberg, Gerald (sys engin'g) [SS]**
88. Weiner, Norbert (feedback) [SS]
89. **West, Gregory (sys allometry) [SS]**
90. **Whiteside, George (sys chemistry) [SS]**
91. **Wilson, Albert G. (hierarchies+) [SS]**
92. Wolfram, Stephen (math sys) [SS]
93. **Wymore's, Wayne (sys engin'g) [SS]**
94. **Yam, Y.B. (NECSI, ICCS conf's) [SS]**
95. **Zadeh, Lofti (fuzzy math) [SS]**
96. **Zeeman (catastrophe theory) [SS]**

ST systems thinking, *SS* systems science, *SiS* systems-integrated sciences that are rather more SS

45.2 Viewing Systems Mega-Domains as on a SPECTRUM

At a Sunday preconference workshop for the International Society for the Systems Sciences (ISSS), attended by a self-organized group of mixed systems engineers and systems thinkers, the 27 participants were given the challenge of listing all of the main workers in any systems area whose work they think should be integrated to form a KB for SS. In 30 min, they produced a list of 56 (nonbold, Table 45.1).

The author of this report was moderator of the workshop, and so did not submit his own candidate names. His additions for this study (40 workers, shown in bold) represent only a partial list of those he considers important. Adding them together (Table 45.1) gives an initial list of nearly a 100 lifeworks that need to be synthesized. This task is one of the official projects of an ongoing Special Integration Group (SIG) of the ISSS and of the aforementioned INCOSE-SSWG whose work is now connected by a memo of understanding between the two professional organizations. There are about an equal number of ST Lifeworks representing ST to SS Lifeworks representing SS and a small number of the new Systems-Integrated Sciences (SiS). These designations are the author's opinion based on the criteria listed in Sect. 45.4 and would probably be challenged by a number of the authors. But that is precisely the point of this chapter. Many are claiming their work to be SS when it is not according to our criteria. It is clear why they do this (wanting to capture the reputation of the sciences, or the major funding of the sciences; wanting to render their work more rigorous; wanting to justify greater efficacy of their work, etc.). So these should be considered just an initial designation to be debated much in the future. Strong math-based works were counted as SS, and strong management- or human-system-based were counted as ST.

One unexpected use of this list is to enable a modest "shock and awe" reaction. It could be used by anyone who feels they are a "systems specialist" or expert to test themselves (as systems engineers and sustainability workers must be by name alone). Can I recognize all the workers? Am I familiar with their work and how it relates to systems? And please remember that this is only an initial, partial listing.

Perusal of this one sample suggests how great is the potential span of available information for systems approaches, whether ST, science, or anything in between often defying simple, exclusive classification. Analysis indicates that several of the workers might be grouped together in what might be called "clusters" or "cohorts" of similar coverages, goals, or techniques. The type of information characteristic of each of these clusters in the sample is very different. Placing just some of these names into "alike" work gives an interesting set of cohorts as shown below that could also be used by any self-proclaimed systems expert to do a self-test of which group output they are familiar with and which not. It is an invitation to more learning to all of us and a challenge to the project on unification of SS of the INCOSE-SSWG.

- *SYSTEMS THINKING MEGA-DOMAIN* (less direct use of scientific method and/or natural science expt's; focus or reliance on logic, philosophy, or human application attempts)
  - Original and new systems dynamics (Forrester, Meadows, Senge, Cabrera, MIT)
  - Soft systems methods (Checkland, Flood, Jackson, Midgely, Warfield)
  - Systems management (Ackoff, Ashby, Beer, Churchman, Mitroff)
  - Models based on engineering per se (Wymore, Iberall, Ring, Lloyd)
  - Systems Philosophy (Bunge, Bahm, Laszlo, Bateson, Rousseau)
- *PROTO-SYSTEMS-SCIENCE MEGA-DOMAIN (exhibits more use of scientific method; comparison of and abstraction across evidence from natural science exp'ts or based on math formalisms) (isomorph means constant and similar abstract patterns of structure or dynamics across the systems of many different domains or disciplines when compared)*
  - Original work on GST and Cybernetics (vonBertalanffy, Boulding, Gerard, Weiner)
  - Mathematical systems theory (Klir, Rosen, Rapoport, Thom, SysAnalysis/OR)
  - Isomorph/or/theory based on science results
    - Models using mostly one systems isomorphy (Prigogine Mandelbrot, per Bak, Corning, Weiner, Thom, West)
    - Models using four or more systems isomorphies (Forrester, Wilson)
    - Models using several systems isomorphies (Miller [LST, Living Systems Theory], Odum, Auyung)
    - Models using a very large number of isomorphies and their influences (Troncale)
  - New systems-integrated natural sciences (much less focus on isomorphies; more on evidence, but whose evidence increases systems understanding)
    - Systems biology (Mesarovic, Allen, Hood)
    - Systems chemistry (Karplus; Whitesides)

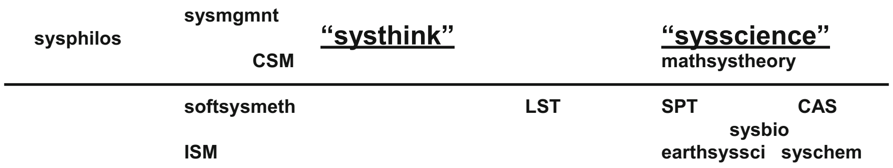


Fig. 45.1 A simple spectrum of systems approach areas from more human to more evidence-based ST and SS, shown in slightly larger font size as clusters on the single spectrum of development

- Earth SS (Lorenz)
- Complex adaptive systems in physics (many institutes and centers)

We prefer to look at these distinct "areas" as on a spectrum with characteristics and measures varying gradually across the spectrum. In this view, as shown in Fig. 45.1, ST is toward one end of the spectrum and the natural SS toward the other end. This perspective has the advantage of placing both "thinking" and "science" on a level of shared characteristics and not so much as in opposing camps as the historical traditions might suggest.

The reader may want to adjust this spectrum by moving the domains left or right, but based on what? The home cluster of the reader? Their position in the science wars? [1, 2]. Developing the criteria to guide a consensus on where to place what on this spectrum is an important challenge to the aforementioned INCOSE-SSWG project. For the purpose of this chapter we ask, . . .is all this information across the spectrum truly science-based, which is to say evidence-based? And, as is so often said, why should we care?

45.3 Juxtaposition: Science and Engineering – Three Short Tests of Whether or Not ST Is SS in SE

All words are promiscuous. Just look at a dictionary. Most words have four or five meanings. Of all words, science might be one of the most loosely used with systems close behind. If a field of study or application has the word "science" appended on it, it probably is not science as scientists would define the word. Biology, geology, physics, chemistry – all normally do not have the attachment "science" – and all are indisputably versions of science. But are the widely recognized specialties of design science, management science, behavioral science, and many more, truly science? Further for this article, can their systems-focused counterparts be called SS. Numerous discussions with members of these disciplines show each strongly defends their claim to be science. But even in the sciences, there is a pecking order. Physics is the hard science and some may even permit chemistry in that category. Biology and geology were dismissed as mere descriptive specialties in the beginning although now they can clearly claim to be natural sciences. Presumably, this would allow results from them to be called a science of systems. But let us explore that assumption.

45.3.1 What Is Science? What Is Engineering? What Is Their Interrelationship?

For the purpose of this chapter, science = use of the scientific method. It has many versions. A recent concept map made by a group of SEs debating the above three questions via SSWG did not look like the science my colleagues and I in cell and molecular biology practiced. Our experimental process involves previous findings, induction of a hypothesized causal chain or alternative mechanisms, isolation of variables by establishing controls, design of an experiment testing the hypothesized mechanism, use of established or innovative techniques and tools to measure experimental outcomes, statistical analysis of measures, coupling conclusions very tightly to the original hypothesis, iteration, reproducibility, and so on. And then there is math, a strong companion of science that does not use experiment at all. Math uses the rules of symbol transformation for each math to discover new relations never before seen. It is truly fascinating that math sometimes predicts relations very long before they are experimentally demonstrated.

Many of the SE-SSWG discussants have strongly protested that engineering is very different from science. Indeed, engineering has much overlap with human need and so has a more anthropocentric orientation than science. You might say engineering is science with a human purpose. By contrast, reference to purpose and goal is generally forbidden in the natural sciences. They introduce the shadow of a divine designer, or human subjectivity, and interpretation into what is supposed to be a completely neutral and objective questioning of how nature works. Yet many recognize that much of engineering is based on the results of science. Chemical engineers use chemistry, aeronautical, mechanical, and space engineers study and use a great deal of physics and math, and so on. Our cars, planes, and phones work reliably because of an engineer's clever use of science. Also, engineering has a central place for testing and evaluation just as science does. So their Venn circles overlap, but not completely.

Presumably, systems engineers would need to study SS just as other engineers study the basic science that is the foundation for their applications (as in chemistry for chemical engineers). But what if they only study ST that we discriminate from SS? Because SEs in many corporations were needed because of the sheer size and complexity of modern engineering projects and products, many SEs study and daily use various systems management and OR tools, techniques, and KB. This is clearly needed. Do these aspects of ST include SS? Would SS provide additional and significant prescriptive information about the "systems that are built" in contrast to how to "organize humans best to produce a system?" Let us quick test this in three ways.

45.3.2 Spot Test One: Recent SE Papers on ST and SE Systems Education

Recently, there has been a flurry of papers on teaching ST for SE education in SE-related publication venues [4–8]. How do these papers portray ST versus SS?

Arnold and Wade of the Stevens Institute [4] observe that there is no widely accepted definition of ST, cite and critique those in use, and propose a new one that primarily adds the concept of "purpose" to the features of existing definitions. Their paper clearly describes the importance of systems awareness in solving today's complex systems-level problems. They quote Forrester that "ST implies a rather general and superficial awareness of systems," but then conflate it with the SS that this chapter argues could yield a much more detailed and rigorous awareness of systems if adopted by SEs. Although they use ST interchangeably with SS, citing SS only three times, they never define SS as distinct from ST. In fact, they even reduce ST entirely to that practiced by the Systems Dynamics Society and MIT. This ignores a great deal of the literature of the ST mega-domain much less the SS mega-domain. They only cite 4 of the 46 names in ST in our Table 45.1 and zero of the 50 SS lifeworks. They only cite about 5 of the 110 isomorphic systems processes of the SPT (which is intended as a nascent SS). They do not cover the differences between the use of "purpose" or "goal" in human ST relative to its much different counterpart in SS, "function." This chapter provides evidence that ST and SS are conflated in the SE literature, and as it is coming from the Steven's Institute of Technology, a strong SE provider, it indicates that not only is SS neglected in SE, but that even ST is poorly covered.

Camilla and Ferris of the University of South Australia and the Center for SE, Cranfield University, have written four papers on the evaluation of ST competencies in SE education [5–8]. Their work is thorough and focuses on a much neglected aspect of both SE and general education assessment – the affective domain relative to what is cognitively learned. Although their papers also draw attention to the need for a consensus definition of ST, suggesting another new definition, one of the four does not mention SS even once (while it does ST nearly 100 times), and the other three of the four mention SS only rarely (4, 0, 1, and 0 times). Across all four papers, they cite ST 258 times, but cite SS only five times. In all of these cases, the usage is in historical contexts; that is, they use SS/ST conflation as some of the historical figures did. Unfortunately, many of the so-called guru's of systems themselves conflated SS and ST, in our opinion erroneously. Camilia and Ferris [5] do summarize the entire ST mega-domain admirably. Of their more extensive reference set of 88, compared with our Table 45.1, they cite 18 of our 46 ST lifeworks but only 3 of our 50 SS/SiS lifeworks and for those only the aforementioned conflating guru's. So while serving a very useful purpose for ST and SE, this paper series neglects SS input as does most of conventional SE. In their conclusion, they state "Our review of the curriculum of 33 institutions in the U.S. and Australia . . . shows that most institutions do not offer their students a specific ST course." This author's survey of the same source indicates that none offer a course in SS as defined here and not conflated with ST. Perhaps the origin of the ST–SS conflation, this chapter argues against can be found in the comparative ignorance of SS by the SE field relative to its dependence solely on ST.

45.3.3 Spot Test Two ST: INCOSE Pioneers as a Case Study

INCOSE recognized Checkland (2008) and Warfield (2007) as "Founders." Were they systems thinkers or systems scientists? Both were also Presidents of the ISSS (an MOU Collaborator with INCOSE), at the same time as this author was Vice President and Managing Director, so he interacted with them frequently. Numerous conversations revealed these workers (and other STr's like Ackoff, Churchman) as strongly ST-based, even opposed to natural science. This is supported by recent summary of ST (such as [4]), where only about 5 of the 110 systems mechanisms of SPT [11–15] are recognized in ST relative to the SPT/SP/SSWG project. That paper's "functionalist" category is defined as science would be, but groups assigned to it are neither science nor SS according to our criteria. ST has many mechanisms unrepresented (network theory, fractals, natural recycling, self-criticality, and symmetries). This would make the ST KB very much weaker and less prescriptive than SS on a general theoretical basis. Yet ST workers continue to cite themselves as SS workers. A quick look at Wikipedia indicates that this conflation of theoretical focus (thinking vs. science) is widespread. For example, Warfield's last major opus was entitled, *Introduction to Systems Science* [10] showing that he regarded himself as both a systems thinker and a systems scientist. Unfortunately, many in ST do. Meanwhile, for this chapter, workers strong in mathematics – as was INCOSE Founder W. Wayne Wymore – are midway between ST and SS, but because of their strong, necessary emphasis on applied engineering, lean heavily to the ST side.

45.3.4 Spot Test Three: Systems Engineering Introductory Texts

The SPT that purports to explain how many successful natural systems work using ~110 very defined systems processes and 100s of linkage propositions between them [11–15, 18] is the framework SSWG SPT/SP is using to integrate our currently fragmented systems sources (Table 45.1). I examined the index of the latest edition of the SE text by Blanchard and Fabrycky [19] used in many SE core courses. The test was to see how many of the SPT patterns, structures, principles, or universal isomorphies (systems processes) were represented in the text. I found brief or tangential mention of five, and more coverage of five others, but completely from a human system or corporation point of view. For example, feedback and control are the best covered, but the discussions of input–output, networks, and purpose are restricted only to corporate coverage. A good example is cycles and cycling; it is covered only in terms of the product(ion) life cycle without any information from the many cycles studied in nature or the immense number of details we have learned from those studies (see [15] for 72 case studies). In summary, compare only human-based coverage of ~5 with 110 in the SPT plus the 100s of LPs (Linkage Propositions). In fact, the entire spectrum of systems areas cited above was very poorly covered in this fundamental preparation for many SEs. Only 3 of the 96 sources in Table 45.1 were cited. Perhaps a dozen pages covered all of systems theory (as a poor stand-in for SS) at a level that was outdated 50 years ago. A text I would prefer that is still limited would be Hitchins or the new text by Mobus and Kalton [20]. These texts cover more isomorphic systems processes in greater depth. But still, the emphasis in all three basic texts used to educate the new generation of SEs very poorly represents SS and is entirely oriented to just one side of the spectrum, ST. Although this is entirely understandable and targeted for SEs, it leaves out rather completely what can be learned from how natural systems have solved the problems of complex systems across nearly 14 billion years of trials and optimization. What ST lacks is the "prescriptive" that is based on scientific evidence and testing in trillions of trials or more.

45.4 Minimal Criteria: For the Scientific Method, a GST, and for a Science of Systems

Careful study and selection of criteria for identification and discrimination may be the best start for distinguishing areas like ST and SS. The founders of GST never did this adequately. Yet, it has proven best in the past for such a highly interdisciplinary group as us to debate until a consistent list of criteria are reached before arguing fine points of competing theories and approaches. Criteria can help focus discussion on key issues, improve communication though common terminology, and provide a basis for judgment on inclusion or exclusion in the mega-domains. At INCOSE-SSWG International Workshop sessions and ISSS SIG (**S**pecial **I**ntegration **G**roup) sessions, the following criteria lists for a general systems theory and for a science of systems were suggested, but consensus was not yet reached.

Criteria for a scientific theory: (1) vast KB consistent with past evidence usually on nonhuman systems; (2) hypotheses of alternative, falsifiable causalities; (3) prediction(s) from hypotheses; (4) control of variables; (5) experiment to test hypothesized causality; (6) statistical analysis or often involves measurement (empirical); (7) limit conclusion to experiment; (8) cycles of iteration and recursion; (9) evidence so confirmed provides explanatory power; (10) demonstrated predictive power and fecundity; (11) tested by many independent investigators and methods; (12) requirement of replicability; (13) consistent with a wide range of other disciplines (unity and fit); (14) parsimonious; (15) consensus agreement of experts on conclusions; (16) results are correctable; (17) inductive; (18) results discipline use of language or construction of models; (19) often requires invention and calibration of new measurement instruments and tools; (20) subsumable; (21) quantifies uncertainty; (22) characterized by strict documentation, archiving, and sharing; (23) focuses on phenomena; (24) focuses on natural processes; and (25) or/involves or enables allowed math formalism transformations. Theories are not laws but may contain several laws. Discovery science and experimental science utilize slightly different methods. This is merely a summary, but it gives some basis for deciding whether a practice is ST or uses the science above to explain systems in a scientific manner.

Many of the systems-associated disciplines, old and new, that describe themselves as "science" simply do not have all of these characteristics (e.g. design science, social science, behavioral science).

Criteria for a general systems theory included: (1) focuses on finding isomorphisms (patterns consistent across many specific and real instantiations of systems); (2) may be of processes, structures, principles, or patterns; (3) protocols for valid abstraction; (4) maps to real systems; (5) high level of abstraction from particulars; (6) has minimal set of isomorphisms defined; (7) describes how the isomorphic processes interact to explain how systems work; (8) can also describe how systems do not work; (9) describes how systems come into being (origins); (10) explains how a systems fulfills a function or purpose; (11) explains taxonomy of types of systems; (12) provides rationale for identification of system boundaries; (13) defines extents and limits of application; (14) supports or enables modeling and simulation of any real system; and (15) must provide evidence of presence of isomorphy across very wide range of types of system. This summarizes a medium consensus between Hybertson and Troncale, reached on October 1, 2014.

Criteria for a science of systems (SS) then would include an amalgam of both lists: the scientific method applied to GST or scientific realism [9].

45.5 Discrimination: Similarities and Differences Between ST and SS

Although the SPT/SP projects of INCOSE-SSWG would defend the need for ST, it suggests expansion of awareness to SS. Initial lists of some of their similarities and differences might increase understanding of not only how they are distinct, but also how one could join with the other to employ ALL of the spectrum of systems knowledge to SE.

45.5.1 An Initial List of **Nine Similarities***: When "Systems-Level" Is Added to Normal "Science"*

ST and SS share these similarities to reflect and build upon: (i) Both have the *same universal processes* at work in their systems. I have personally argued this point with many ST gurus (Ackoff, Churchman, Jackson, Checkland, etc). Most believe/teach (often only implicitly) that human systems are entirely different from natural systems. Yet to SS folks, human systems, whether individual, group, or nation, exhibit "hierarchies," "cycles," "self-organization," "feedbacks," "self-criticality," "equilibria," "flows," "fractal structure," "chaos," and more as demonstrably as natural systems. How can they maintain that these are different on the systems level? (ii) Both encounter the *same messy problems*, that is, limits on complexity, need for adaptation, dealing with chaos, using chaos, limited resources, and many more. (iii) In the SPT, there is a *unbroken continuity of origins*, that is, the chasm between human and animal origins, between the various scales of physical systems, has consistently been bridged by scientific investigation and increased understanding, never the opposite. They are one out of (from) the other instead of one distinct from the other. That is probably why they share the universal processes. (iv) The ultimate success of both is due to dynamic stability and *sustainability* relative to our space:time configuration. (v) Both use the "*comparative*" inquiry style relative to less use of comparative in conventional science. (vi) Both have special need to explain *complexity, emergence,* and *chaos*. (vii) Both exhibit central role of *nonlinear causality*. (viii) Both exhibit a central role for *networks*. (ix) *Applications:* the successful case studies of numerous natural phenomena should be considered "applications" even though they are not human.

45.5.2 An Initial List of **Fourteen Dissimilarities**

But ST and SS have these differences that may have to be overcome or used as a basis for complementary enhancement: (i) *Main subjects of study:* are different as ST focuses almost exclusively on humans and their institutions or societies; SS on natural phenomena. At the level of particulars, there is no question of difference with natural systems, even biological. The similarities are only revealed in abstracting from, releasing obsession with particulars, to compare how the particulars interact. But it is very difficult for humans to accomplish this release from particulars. (ii) *Measurability:* is much easier in natural systems than humans due to distance from the measuring scale. (iii) *Methods and Tools*: study of natural systems has resulted in extensive innovation of measuring devices and techniques (we need to list everything from microscopy to telescopy to spectroscopy and more) that cannot be applied to human behavior. (iv) *Terminology* is quite different because specialties studying human and natural systems have been separated so long. This difference is hard to overcome. (v) What is of *Significance and Value* in each domain of systemness is different. (vi) *Purposes:* ST seeks to improve human systems; SS to research and understand natural systems phenomena as function. (vii) *Uses*: ST emphasizes direct applications, while SS contributes to basic research or theoretical understanding. (viii) *Scientific Method*: SS requires rigorous application of the protocols of natural science, while ST cannot readily achieve this. Guarantor's of truth are dramatically different, (ix) *Facts, Results, and KB*: SS collects, organizes, archives, and curates vast quantities of experimental results called facts; many proponents of ST argue that facts do not exist. (x) *Theory Source:* ST comes from past attempts to solve human systems problems; in SS from evidence derived from comparing natural systems experiments. (xi) *Objectivity–Subjectivity:* It is easier to test natural science systems in an objective manner, while we are the subject of sociopolitical and engineering studies. (xii) *Determinism* versus *Free Will*: Humans are capable of completely ignoring prescriptions, while nonhuman systems cannot. (xiii) *Anthropocentric versus Ananthropocentric:* ST versus SS. (xiv) *Limited Range of Scales versus All Scales:* again ST versus SS when human systems are regarded as natural.

45.6 Scenarios for Cooperation Between ST and SS

What can we build on to bring these two camps together to help each other in true complementarity? If they are on the same spectrum, they should be able to augment each other to produce a hybrid that has much greater strengths, advantages, and wider application than each alone. Here are five ways that they might build on each other rather than oppose each other.

(1) *Focus on evidence-based prescriptions through testing: "Translate" from human to natural systems:* the SPT/SP projects of the SSWG suggest these strategies for synergy relative to testing: (i) Engineering places a very high value on testing and evaluation. Recognition by systems thinkers that SS is backed up by peer-reviewed experiments, and by only selecting commonalities when widely different disciplines, phenomena, and scales are compared, may convince some that there is a high degree of testing and evaluation to the SPT prescriptions. (ii) Some progressive human systems thinkers may concede that it is easier to isolate causal influences in natural than human studies. But it is also easier to study chaos and emergence or other nonlinear phenomena in natural systems. Again testing wins out. (2) *Focus on problems needing solution:* Both try to solve system-of-systems problems, so can use SPT as a common focus. The linkage propositions between the systems processes lift systems theory to the meta-level of detailed explanation of how systems work thus providing a tool to address the complexity issue faced by engineering. How to make systems adaptive is a modern problem for engineers (as is how to handle emergence). Natural systems have encountered and solved this across 14 BYs (see recent biomimicry, systems mimicry Natural Systems Working Group Webinar). (3) *Focus on isomorphic systems processes and pathologies* and their multiple simultaneous influences. SPT is based on

recognizing the isomorphic systems processes that lead to universal patterns in both natural and human systems when studied on the abstract level. They demonstrate the same systems archetypes or architectures in SE terms. This gives much hope for ST moving to SS. The SPT/SP framework and synthesis provide an unprecedented level of detail in its 110 SPs and 100s of LPs to explain how systems work for SEs, its spin-off, systems pathology [18], recognizes many systems dysfunctions not known in SE. (4) F*ocus on modeling and simulation:* If SSWG is able to verify a general model of models, it may be of practical use for many specific models made by INCOSE-SE and MBSE-WG (Model-Based SE). (5) *Vet alternative frameworks for synthesis*, *integration, and unification* to enable the synergy that is likely between a widened ST perspective and SS.

45.7 Summary/Conclusion

What we need is a list of consensus criteria on what constitutes "science" or not to use in distinguishing thinking from science as regards systems-level awareness and approaches. Arguing over and coming to agreement on individual criteria to include is less partisan and more amenable to compromise than arguing directly about the designations. This chapter is a call for a "criteria" debate across both the thinking and science communities that would lead to a mutual consensus and communication.

A useful and convenient surrogate for all of the above is to use the extensive peer-reviewed literature of the several natural sciences (astronomy, physics, chemistry, geology, biology, computer science, and mathematics). Results reported in these for a very wide range of nonhuman phenomena, when properly abstracted, would necessarily exhibit all of the above criteria and yet be evidence for how systems work or do not work at a very fundamental level (the level of isomorphisms). The extensive list of such "proven design" isomorphies in prototype SS's (such as SPT) [12–15] would provide a valuable, list of specifics as a checklist for SE designs and testing. A further "failsafe" is the demand that these abstractions be compared for similarity and consistency. It is likely that an observed process or pattern proven across many natural science disciplines and phenomena, observed in a wide range of scales of system, originating at widely different times, by different mechanisms, across disciplines, studied by independent and differing tools and techniques, and different investigators would hold true for improving SE designs.

References

1. Otto S (2016) The war on science: who's waging it; why it matters; what we can do about it. Milkweed Editions, Minneapolis
2. Mooney C, Kerschenbaum S (2009) Unscientific America: how scientific illiteracy threatens our future. Perseus Books, New York

3. Troncale L. Webinars for INCOSE CxSWG; INCOSE-Fellows; INCOSE-CAB; INCOSE SSWG; INCOSE-NSWG over last nine years (access INCOSE Website for Webinars) plus presentations at INCOSE IW's, IS, CSER, LA and SD Chapters.
4. Arnold RD, Wade JP (2015) A definition of systems thinking: a systems approach. Procedia Computer Science 44:669–678
5. Camelia F, Ferris T (2016) Systems thinking in systems engineering. 26th Annual INCOSE IS, Edinburgh, Scotland
6. Camelia F, Ferris T (2016) Validation studies of a questionnaire development for students' engagement with systems thinking. IEEE Transactions on Systems, Man, and Cybernetics: Systems:1. Accepted. Preprint
7. Camelia F, Ferris T (2016) Undergraduate students' engagement with systems thinking: results of a survey study. IEEE Transactions on Systems, Man, and Cybernetics: Systems:1. Accepted. Preprint
8. Camelia F, Ferris T, Cropley DH (2016) Development and initial validation of an instrument to measure students' learning about systems thinking: the affective domain. IEEE Transactions on Systems, Man, and Cybernetics: Systems. Accepted
9. Rousseau D, Billingham J, Wilby JM, Blachfellner S (2016) The synergy between general systems theory and the general systems worldview. Systema: Special Issue - General Systems Transdisciplinarity 4(1):61–75
10. Warfield JN (2006) An introduction to systems science. World Scientific Publ, Singapore, p 403
11. Troncale L (1978) Linkage propositions between fifty principal systems concepts. In: Klir GJ (ed) Applied general systems research: recent developments and trends : N.A.T.O. Conference series II. Systems science. Plenum Press, New York, pp 29–52
12. Troncale L (1982) Linkage propositions between systems isomorphies. In: Troncale L (ed) A general survey of systems methodology: Vol. I. Conceptual and mathematical tools. Intersystems Publ, Seaside, pp 27–38
13. Friendshuh L, Troncale L (2012) SPT I.: identifying fundamental systems processes for a general theory of systems (GTS) Proceedings 56th annual conference., (ISSS), San Jose, California, (Go to http://journals.isss.org/index.php/proceedings56th), ISSN 1999-6918
14. McNamara C, Troncale L (2012) SPT II: how to find and map linkage propositions for a GTS from the natural sciences literature. In: Proceedings of the 56th annual conference, (ISSS), San Jose, California, p 20. (electronic proceedings: Go to http://journals.isss.org)
15. Troncale L (2012) Proving isomorphy: 72 tests for "cycles" across disciplines, domains, and scales. lecture at ISSS'12. In: Proceedings of the 56th annual conference, San Jose, California, ISSS. Go to http://journals.isss.org/index.php/proceedings56th)
16. Sillitto H (2012) Integrating systems science, systems thinking, and systems engineering: Understanding the differences and exploiting the synergies." *Thales*. Personal communication; Permission granted to INCOSE to publish and use
17. Jackson MC (2009) Fifty years of systems thinking for management. University of Hull Journal of the Operational Research Society 60:S24–32
18. Troncale L (2011) Would a rigorous knowledge base in systems pathology add significantly to the systems engineering portfolio. In: CSER'11 proceedings, conference on systems engineering research, (pp. 1–17). And (2013). Systems processes and pathologies: Creating an integrated framework for systems science. INCOSE International Symposium 23(1):1330–1353
19. Blanchard BS, Fabrycky WJ (2013) Systems engineering and analysis, 5th edn. Prentice-Hall Int'l Series on Ind&SE, Pearson
20. Mobus GE, Kalton MC (2015) Principles of systems science. Springer, New York, p 755

Chapter 46
Three General Systems Principles and Their Derivation: Insights from the Philosophy of Science Applied to Systems Concepts

David Rousseau

Abstract Systems Engineering is currently largely based on heuristics, but it is increasingly recognizing that it needs a scientifically profound systems theoretical underpinning that would enable it to deal with systems in a more holistic manner. Such a scientific general systems theory does not yet exist. The principles-laws-theories model of modern science suggests that such a general systems theory could arise from the discovery of scientific systems laws, whose discovery in turn depends on the formulation of scientific systems principles. In this chapter, I discuss the nature of such principles, show how scientific systems principles could be discovered by applying insights from the philosophy of science and illustrate this approach by deriving three general systems principles that have practical significance for science, design and management.

Keywords Systems science • General systems theory • GST • General systems principles • General systemology

46.1 Introduction

The complexity of the systems being developed by systems engineers is increasing without limit [1], and this trend is starting to strain the heuristic methods of contemporary systems engineering (SE). Many systemic processes have been studied scientifically [2], but SE has no theoretical foundation for characterising systemness as such [3]. This shortcoming has eroded systems engineers' ability to predict the outcome of design decisions [4] and left them without a principled basis against which they can check unexpected system behaviour [3]. In response, the *International Council on Systems Engineering* (INCOSE) identified 'expanding the theoretical foundation for systems engineering' as one of its six imperatives for SE

D. Rousseau (✉)
Centre for Systems Philosophy, Surrey, UK

Centre for Systems Studies, University of Hull, Hull, UK
e-mail: david.rousseau@systemsphilosophy.org; d.rousseau@hull.ac.uk

in the next decade [1]. In its *Systems Engineering Vision 2025* INCOSE explains that: ‘It is therefore important to develop a scientific foundation that helps us to understand the whole rather than just the parts, that focuses on the relationships among the parts and the emergent properties of the whole. This reflects a shift in emphasis from reductionism to holism. Systems science seeks to provide a common vocabulary (ontology), and general principles explaining the nature of complex systems’ [1].

Note that the request here is for principles that ‘explain’, rather than *describe*, and moreover that explain ‘the nature of’ complex systems, rather than explaining *how* complex systems *work*. In addition, the call is for a ‘principled basis’ against which to check empirical outcomes in SE design, so the envisioned principles must be such that they engender a systems theory that is scientific, general and quantitative. In this light, the sought-for principles would be scientific and fundamental in a way not fulfilled by the heuristic principles in the current systems literature (which although useful are only descriptive and qualitative) [e.g. [5, pp. 17–29], [6, pp. 33–38], [7, pp. 60–69], [8, pp. 99–105]]. The sought-for ‘principled basis’ would be explanatory, general, and quantitative in a way not fulfilled by current scientific theories about systemness (e.g., Troncale’s ‘systems processes theory’ (SPT) [9–11], which is qualitative and phenomenological,[1] or by modelling approaches (e.g., Odum’s ‘energese-based ecology model’ [12], which is grounded in analogies with electrical laws and circuits).

The quest for such a ‘systems science’ originated in the 1930s with Ludwig von Bertalanffy, who called for a general understanding of systems, saying: ‘It seems legitimate to ask for a theory, not of systems of a more or less special kind, but of universal principles applying to systems in general. In this way we come to postulate a new discipline, called General System Theory. Its subject matter is the formulation and derivation of those principles which are valid for “systems” in general’ [13]. Von Bertalanffy held that General Systems Theory ‘is [the] scientific

[1]Von Bertalanffy argued that the phenomenological “systemic isomorphisms” signify the existence of general [systems] principles that would form the foundation of a GST [24, p. 33]). In contrast, Troncale’s “Systems Processes Theory” (SPT) [9–11] replaces the inference to general principles by proposing the existence of “isomorphies” as pre-existing abstract entities that are the causal precursors of the empirical ‘system patterns’ that recur isomorphically across kinds of systems [27, p. 17]. This postulation of abstract entities that have causal powers suggests SPT is a form of quasi-idealism akin to Platonism and hence not fully consistent with Naturalism. Although SPT is scientific, it is not quantitative but grounded in qualitative “linkage propositions” that denote causal relations between the abstract “isomorphies” [11] (e.g., “‘feedback loops’ are a partial cause of ‘cycles’, and “‘symmetry’ inhibits ‘oscillation coherence’”). SPT has valuable explanatory dimensions, but it does not predict isomorphies or explain their variety, instead bringing them into the theory as abstract fundamentals proposed on the basis of a list of empirically observed kinds [58, p. 16] of optimized design patterns [27, p. 17]. The absence of general principles entailing both isomorphies and their variety precludes SPT from being truly general. For example, SPT currently has no basis for showing that a systems model built on SPT is complete, in contrast with the way that, for example, Newtonian mechanics applied to solar system models led to the discovery of previously unobserved planets.

exploration of “wholes” and “wholeness”’ [14]. These claims represent a clear precursor to the call from INCOSE quoted above.

Von Bertalanffy’s advocacy of general systems theory (GST) led to the founding in 1954 of the *Society for the Advancement of General Systems Theory*, renamed in 1988 to the *International Society for the Systems Sciences* (ISSS). Subsequently many heuristic methods for action in systemic scenarios were developed, particularly in Management Science and SE, but limited progress has been made with formulating a GST [15]. Although the case for the existence of a GST remains strong [16], it was, until recently, very unclear what a GST might look like, and how its principles might be discovered [17, 18]. Von Bertalanffy and his ‘circle’ had few practical ideas about how to go about discovering general systems principles [19], and recent reviews confirm that no general systems principles have gained broad recognition [15, 18, 20]. Instead, the practical offshoots of theories addressing isomorphically recurring systemic behaviours or structures (e.g. [21–23]), originally foreseen as forebears of GST [24, p. 33], have gained prominence [25], whereas the ‘systemic isomorphies’ they represent have not been assimilated into a general theory [26, p. 248]. Troncale’s SPT *does* assimilate the known isomorphies into a single theory, but as a systems theory it is phenomenological rather than general because it does not predict isomorphies or explain their variety but adopts them as fundamental entities in line with empirical observations[27, p. 17] (see also note 1). Consequently, the need for a scientific and foundational GST as foreseen by von Bertalanffy remains unmet. This need is however still pertinent, for example, in a workshop organised in 2014 by the National Science Foundation, with support from INCOSE and the Systems Engineering Research Center, GST was identified as an important element of a future systems science [3], and this view is echoed in the INCOSE *Vision 2025* [1].

A historical challenge for this work has been the lack of a broad consensus on what ‘systems principles’ are, and how they would empower a science of systems [5, 28–30]. However, a new initiative launched in 2014 and jointly supported by the ISSS, INCOSE’s Systems Science Working Group (SSWG), the Bertalanffy Center for the Study of Systems Science (Austria), the Centre for Systems Studies in the University of Hull (UK) and the Centre for Systems Philosophy (UK), have done much to clarify the concepts, assumptions, frameworks and potential of GST, and so laid a foundation for systematic work towards establishing a practically significant GST. This activity has so far resulted in two workshops in 2015 sponsored by the INCOSE SSWG, the publication in 2016 of six papers in a special issue of the journal *Systema* [16, 20, 31–34]and a book proposal accepted for publication by Springer in 2017 [35]. The present chapter builds on this foundation, and its aims are to (a) present a principled case for what systems principles are, (b) describe a way to discover systems principles using science and the philosophy of science, (c) demonstrate this discovery method by deriving three general systems principles, (d) explain how knowledge of these principles can have practical value, and (e) briefly outline the scope and potential of needed and feasible future work in this area.

The present chapter exploits the necessary connection between metaphysics and science, a connection that many practicing scientists ignore or dismiss. Science and the technological disciplines usually proceed without taking much notice of the principles that underpin them, but for a young discipline such as systems science, an explicit engagement with metaphysics can help accelerate foundational development and help avoid many of the pitfalls and blind alleys that can confound emerging disciplines. Specifically, it will be argued that if we identify the most fundamental ideas underlying the scientific enterprise, and the most fundamental scientific insights we have about the nature of the world, and use the perspective this yields to analyse our basic ideas and observations about systemness, then we can gain important scientific insights about the nature of systems, and if we bring those insights back into science it will both change how we see the world and create new potential for discovery, innovation, intervention and engineering.

The perspective that will be used here depends on two lenses. The first is a popular metaphysical perspective known as 'scientific realism' (SR). The second is a simple but powerful metaphysical model of the structure of modern science called the 'principles-laws-theories' (PLT) model of modern science [36]. Briefly, SR encompasses three commitments: that the world has a definite and mind-independent structure, that scientific theories are true or not because of the way the world is, and that our best scientific theories are approximately true of the world [37]. From the early twentieth century until nearly its close, SR was a controversial position, but currently most metaphysicians of science endorse it [38, p. 299]. The PLT model represents an early attempt (1996) in the modern resurgence of metaphysics to show how science depends on metaphysical principles, and how such principles relate to scientific laws and scientific theories. The metaphysics of science has advanced rapidly in the last two decades, but in my view the basic structure of the PLT model is still the most practically useful framing we have of these relationships. The PLT model will be described in more detail below.

46.2 Science as Rational Inquiry into Nature

Science represents a rational inquiry into the nature and behaviour of naturalistic things, that is, all things subject to the constraints and laws of nature, thus encompassing both natural and artifactual things. Via this investigation scientists strive to develop theories and models that in their abstract workings reflect the concrete workings of naturalistic systems. This ambition was first explicitly articulated by Galileo [39] and is here illustrated in Fig. 46.1. On the left is represented a naturalistic system proceeding through a series of changes due to causal processes. The right side shows a formal (conceptual) system stepping through a series of propositions via logical inferences. If the initial propositions encode the initial state of the naturalistic system, and the concluding propositions can be mapped back onto the final state of the system, then the conceptual system is a scientific model of the naturalistic system. A scientific model characterizes a specific instance of a

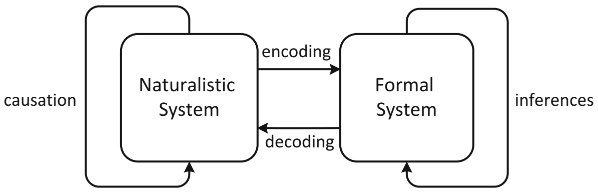


Fig. 46.1 The modelling relationship in science (Adapted from Robert Rosen [40]; image © D. Rousseau 2016, reproduced with permission)

naturalistic system, and if it is generalised to characterise the instance class, then the resulting formal system is a scientific theory of that class of naturalistic phenomena. Figure 46.1 thus reflects the scientific position that naturalistic phenomena (both natural and artifactual) can be explained and predicted in a logically coherent manner.

There is a substantive literature on the distinctive elements of the scientific inquiry method that enables such theories to be constructed in an efficient way (e.g. [41]), but for present purposes I will focus instead on the philosophical underpinnings of the modelling relationship. In order for it to be possible to establish these isomorphisms between naturalistic and logical systems, certain things must be generally true about the nature of the world, and we can see this reflected in the scientific worldview as a set of general assumptions about the nature of the world. According to the PLT model, these general assumptions about the nature of the world constitute the principles of science: they capture fundamental ideas about what is possible or necessary, define key concepts, support scientific reasoning and serve as guiding orientations for doing science [36]. To make this clear I will briefly state two example scientific principles, and discuss some of their entailments to show how important scientific concepts logically arise from such principles.

46.3 Examples of General Assumptions (Principles) in Science

An ideal of science is to minimise the number of its foundational general assumptions (principles).The literature dealing with such attempts in metaphysics is sparse, but it seems possible to reduce the foundational set to about five or so. Many principles stated in the literature are not actually fundamental but are corollaries or combinations of more fundamental ones. Although the foundational assumptions appear simple, they have extensive ramifications, and as their entailments are elaborated they start to overlap, revealing that the principles form a system rather than a simple set. This is not unexpected given that stating specific principles typically draws on concepts established by at least some of the others.

Two examples of such general assumptions are that (a) all changes are caused (the ‘principle of sufficient reason’ (PSR)) and (b) the same causes always produce the same effects (the ‘principle of the uniformity of nature’ (UP)). These principles exemplify the connection between science and rationality because we could not understand it if things 'just happened', and we would not be able to coherently relate effects to causes if there were no consistent relationships between them.

Stated in the form just given the principles are qualitative heuristic principles, and are more 'in the spirit' of science than 'in the mode' of science. However, they are valued in science because they can be framed in terms of relationships between clear and quantifiable concepts, and once this is done, the *scientific* principles they have become can have profound implications for the development of quantitative theories that can make precise predictions. Examples of such 'translations' of principles are given next.

Under scientific realism, we interpret PSR as meaning that every event has an explanation in terms of prior events, such that every event is an effect that was caused by a prior event containing (a) things having the potential to produce the change ('causal powers') and (b) the right conditions to trigger the expression of those powers [42]. Things that have causal powers are 'concrete' (as opposed to imaginary or abstract). Causal powers are kinds of properties that concrete things can have, specifically the power to cause change, that is, to do work. Science has developed the notion of 'energy' to denote 'the ability to do work'. This connection between concrete properties, causal powers, change and energy is very significant because energy is an exact and quantifiable concept in science, and this connection therefore enables us to move from qualitative descriptions in terms of 'properties' and 'change' to quantitative descriptions in terms of energy [43]. This works out in detail, so we can say of something that has a concrete property that, in connection with the associated causal power:

- The *kind* of causal power something has is represented by the *kind* of energy it has;
- The *amount* of causal power it has is represented by the *amount* of energy it has;
- The *kind of change* that it has undergone reflects the *kind of energy* it has gained or lost and
- The *amount of change* it has undergone reflects the *amount of energy* it has gained or lost.

The concept of 'energy' thus functions somewhat like a book-keeping constant for science, allowing scientists to track changes in the kinds and strengths of (concrete) properties as interactions occur. Using the concept of energy as expanded under PSR, we can now see an important entailment of the principle of the uniformity of nature (UP). If the same causes always produce the same effects, then there must be a regularity not only in the sense of kinds of change but also in the sense of amounts of change, so that effects are consistent with and proportionate to their causes. If we express this in energy terms, then UP entails that in causal interactions energy is transformed or exchanged in consistent and proportionate amounts. That in turn entails that in causal interactions, the total amount of energy is conserved. The scientific so-called 'principle of the conservation of energy' (ECP) is thus revealed to be a corollary of the heuristic UP, derived in the light of the concept of energy that arose from the analysis of PSR.

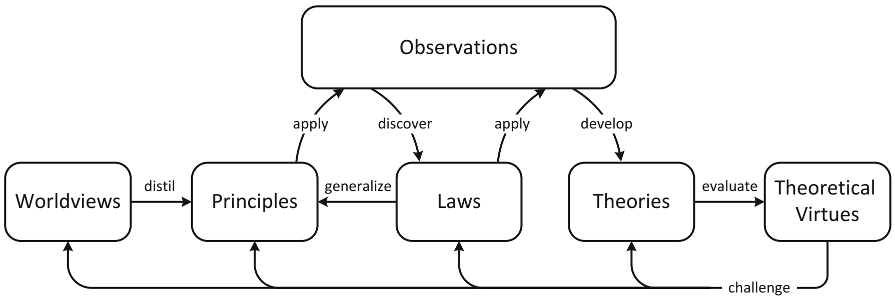


Fig. 46.2 The 'principles-laws-theories (PLT) model' of modern science (Image © D. Rousseau 2016, reproduced with permission)

The brief discussion above well illustrates how scientifically profound concepts like 'energy' can arise from very simple-seeming assumptions such as PSR and UP, and how the application of such scientific concepts can transform the utility of the principles supporting scientific work.

46.4 The PLT Model

Principles, laws and theories interdepend systemically, and this conditions how they are discovered, used and evolve. The 'PLT model' mentioned earlier [36] captures these relationships well, as illustrated in Fig. 46.2 and explained below.

The guiding principles for doing science (e.g., that similar causes produce similar effects) can be distilled from worldviews (e.g., scientific realism). By applying such principles to observations of causal interactions, we can discover laws of nature, which tell us how specific kinds of things change in relation to other changes with given specific contexts. For example, Boyle's law specifies how an increase in the temperature of an ideal gas will cause it to proportionately expand. Conversely, laws can be generalised to suggest new principles, for example, Kepler's second law, which states that planetary orbits sweep out equal areas in equal time, can be generalised to suggest the principle of the conservation of angular momentum. By applying laws to observations of phenomena, we can develop models and theories that explain or predict those phenomena. For example, we can apply Newton's laws to data from astronomical observations to build a theory that explains why we have two ocean tides per day, or to build a model that predicts the occurrence of specific eclipses. In practice, there are often multiple ways of explaining the same phenomena, and competing theories or models are judged as to how 'good' they are by evaluating them against 'theoretical virtues' such as explanatory power, predictive power, simplicity, falsifiability, coherency, empirical adequacy, consistency with well-established theories, etc. [44, 45].

If we cannot develop 'good' theories about a given phenomenon, we question the established laws: perhaps they need additions or refinements, or we need extra ones. To discover new or improved laws, we have to look to our principles, because laws are special cases of how the principles play out in specific circumstances. By making further careful observations of the puzzling phenomena, and then, by strictly applying our principles, we might find better or further laws, which we can then use to develop better theories and models. If we still cannot devise good theories, we question the principles. We can refine or extend them by generalising from laws we already have, or distilling them from the assumptions entailed by our worldviews. If new or improved principles cannot be found, or what we do find does not help us to improve/extend our laws such we can build good theories, then we must question our worldviews, reflecting on how we balance between knowledge, experience and intuitions to find the core beliefs that ground our judgements and actions, and form an adjusted worldview from which we can then adjust or extend our principles, laws, theories and models.

In this way, assessments against the theoretical virtue criteria drive the evolution of theories, laws, principles and worldviews in a systemic way. If systems science is scientific, it will follow the same pattern of discovery and evolution as we search for scientific systems principles, laws and theories. However, to leverage this insight, it is advisable to adopt the methods and language already in use in science and philosophy to model this process and capture its outcomes, so that we can maximise the lessons we can learn from established science and the metaphysics of science, and minimise the effort needed to integrate the findings of systems science back into the established body of science. For example, accepting 'scientific principles' as denoting our most general assumptions about the nature of the world, then entails that scientific 'systems principles' express our most general assumptions about the systemic nature of the world.

In what follows, I will apply these ideas from science and the PLT model to foundational systems concepts, to derive insights for systems science. In the next section, I will prepare for this analysis by briefly rehearsing some basic systems ideas.

46.5 Systems and the Systems Hierarchy

A pervasive idea in systems thinking is that we can arrange naturalistic systems into a hierarchy by sorting things into kinds based on properties that are essential to being members of that kind, and then, ranking them in order of scale and complexity, as shown in Fig. 46.3. The 'layers' are usually referred to as 'levels of reality'. This hierarchy corresponds to the reductionistic idea that systems on one level have as their parts systems from the lower levels, and that properties of the systems at any level can be explained in terms of the properties of, and the relationships between, their parts. There are no unproblematic ways of depicting this hierarchy, but for present purposes, these issues will not matter. For now, it is sufficient to just highlight some basic system concepts involved in constructing such a model.

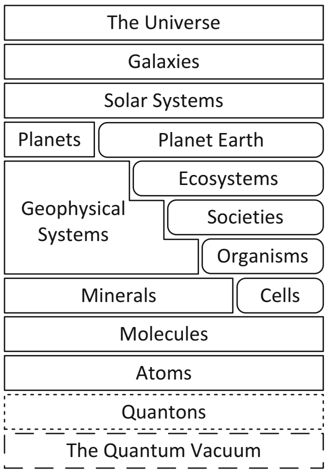


Fig. 46.3 A Hierarchy of Kinds of Systems (Adapted from [46]; image © D. Rousseau 2016, reproduced with permission)

Systems differ from heaps in that the properties of heaps are merely the sum of the properties of the parts, whereas systems have new kinds of properties their parts do not have. These are called ‘emergent properties’, and it is this emergence of new kinds of properties that establishes new kinds of systems [47]. One definition of ‘system’, due to Anatol Rapoport, is that a system is a whole that functions as a whole in virtue of the relationships between its parts [48]. For naturalistic systems, these ‘relationships’ must be concrete and therefore are those established by causal interactions between the parts. In this light, we can understand emergent properties to be new kinds of causal powers that arise due to kinds of causal interactions between parts creating new kinds of wholes. Systems can have a diversity of kinds of parts, and a diversity of inter-part relationships, leading to a diversity of kinds of interactions between the parts. As the diversity of parts, relationships and interactions increases systems are said to become more ‘complex’ [49]. That said, it is important to note that interactions between parts do not always produce new kinds of systems—in fact for the most part, interactions just create new states in existing systems.

These intra-systemic interactions and their consequences can therefore be modelled in terms of two dichotomies. On the one hand, we can characterise intra-systemic interactions in terms of their complexity, by considering whether the diversity involved in the interactions is low or high. On the other hand, we can characterise the interactions in terms of the kind of change they produce, specifically whether they produce state changes in a given system or produce a new kind of system through property emergence. We can show this diagrammatically as indicated in Fig. 46.4.

In the bottom half of Fig. 46.4 (Q4 and Q3), we have interactions that merely produce state changes, either via mechanical interactions within simple systems (e.g., collisions between balls on a billiards table as shown in Q4) or mechanical processes within complex systems (e.g., the processes in a wind power generator as shown in Q3). Although these interactions involve systems, these kinds of interactions are really the business of science as normal—it is essentially mechanistic

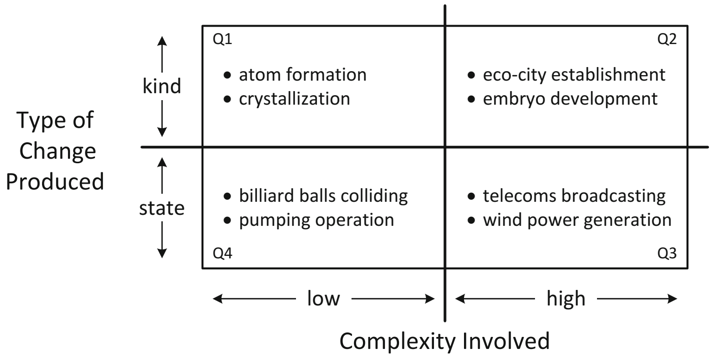


Fig. 46.4 Four Categories of Systemic Interactions (with examples) (image © D. Rousseau 2016, reproduced with permission)

science, and nothing profound is going on from a systems science perspective. In the top half of Fig. 46.4, we have interactions that produce emergent properties and hence produce new systemic identities, for example, when a small number of physical particles combine to form an atom (Q1), or when a large variety of systems combine to establish an eco-city (Q3). Interactions that produce new kinds of systems form a very distinct concern from that of ‘classical science’, and in this area, the systems thinking approach (see e.g. [50]) could make an important contribution. It is important to note that it is not the case that normal science does not take an interest in the origins of new kinds of things, but rather that its approach is predominantly reductionistic, taking new kinds of things to be largely explicable in terms of special states of collections of lower-level entities and so on ‘all the way down’ to fundamental particles (quantons). This was famously expressed by Steven Weinberg saying ‘all the explanatory arrows points downwards’ and remarking that this is ‘perhaps the greatest scientific discovery of all’ [51]. Systems thinkers are typically sceptical about this because living systems exhibit properties that are categorically different from physical ones, such as subjectivity and anticipation, and context can powerfully influence developmental processes, as seen in cultural inheritance. However, it has proved a challenge to formulate a scientifically profound systemic alternative to reductionism (despite a vigorous philosophical debate on this subject (e.g. [52–54])).

Given these ideas about the relationship between properties and interactions in systems, and the ideas previously discussed connecting ideas about properties with causal powers, causal interactions and energy, it is now possible to explore connections between these sets of ideas. For example,

- How does our thinking about emergence and the formation of new kinds of systems change when we interpret properties and interactions in energy terms?
- What can we learn by considering complex vs. simple interactions in terms of energy and conservation principles? and
- What can we learn from this that could make a difference to how we do science and engineering?

46.6 Emergence and the Conservation of Energy

In the first instance, consider a simple scenario involving a low diversity of parts and low diversity of interactions making a new kind of thing. One of the simplest such cases is when protons and neutrons combine to form an atomic nucleus, as happens in stars and supernovae. The atom has properties the parts do not have, being a stable structure that has different causal powers to those of the parts, and different quantities and arrangements of such parts result in different levels of atomic stability. This stability is a new system-level property that emerges as the atom is formed. It is a concrete property, making a difference in casual interactions and therefore is a causal power. However, as explained earlier, causal powers can be represented by energies, entailing that atoms have special kinds of energies that unbound nucleons do not. By conservation of energy, this must have come from somewhere. Given that the emergent property exists due to the interaction of the parts, it seems likely that the parts have given up some of their energy and hence have undergone a reduction in their own properties and casual powers. This is not implausible, as several systemists have pointed out that systems are not only more than the sum of their parts but also *less* than the sum of their parts due to part-properties being constrained by their systemic context. There is even a term for such loss or strength reduction of part properties: 'submergence' [47]. A suggestion that emergence is accompanied by submergence is therefore not in itself new. However, the present claim goes further in two important ways. First, submergence is now *expected*, on principled grounds, rather than just being observed. In science this is called 'retrodiction' and is an important step towards building a theory with predictive powers. Second, because this claim is being made in a scientific way, it can be empirically checked in a precise way. In effect, a qualitative aphorism, that emergence is accompanied by submergence, has been rephrased as a precise quantifiable scientific proposition, namely that the energy gained as the emergent stability property of the atom will be exactly matched by some kind of energy lost to the nucleons through property submergence.

When this is checked for atoms, the expected result is found: the mass of an atom is less than the sum of the masses of its nucleons in their unbound state. This is known in nuclear chemistry as the 'mass defect'. Given the relationship between mass and energy as expressed in $E = \mathrm{mc}^2$, we can calculate the amount of energy this lost mass represents, and indeed it is found to be the exact amount known as the 'binding energy' of the atom, which is the energy that would be needed to break the atom up again.

I postulate that this illustrates a general case, and so propose a first systems principle (SP1): the energy associated with the emergent property in system formation is exactly matched by the sum of the energies lost by the parts participating in that interaction. In systems terms, I call this principle 'conservation of properties', and paraphrase it to say that 'emergent properties are exactly paid for by submerged ones'.

This principle has immediate value for scientific research, indicating that to prove that a property is an *emergent* one rather than a merely unexpected one, one has to demonstrate its balancing interplay with submergence. If this interplay cannot be demonstrated, then the researcher might consider that perhaps the boundary has been drawn incorrectly (so that the allegedly emergent property might be due to parts in advertently left out of consideration), or the essential nature of some of the parts has been misunderstood (so that the property was already present in the parts and not due, at the system level, to systemizing interactions). This finding is an example of GST making a novel contribution to the epistemology of science. Investigating hypotheses grounded in this insight could accelerate progress in boundary science, for example, in the exploration of how properties such as life and consciousness might be emergent on the systemic complexity of lower-level structures. It also has relevance for the analysis of system-level behaviours in engineered product prototypes. This principle also has value for interventions and designs, suggesting that system degradation involves not only loss of systemic functionality but also the correlated re-emergence of previously supressed behaviours of the parts. In studying systems with a view to planning interventions, it is therefore important to find out not only what the system presently does but also to understand what the parts do not presently do but are capable of if ungoverned by systemic constraints. Such attention to the possibility of part-property re-emergence may help to reduce unintended consequences in interventions.

46.7 Emergence and Super-Systems

The first systems principle suggests another one as follows. Systems hierarchy diagrams of the sort shown in Fig. 46.3 illustrate how system levels scale with size and complexity, but this somewhat obscures the fact that it represents a containment hierarchy, such that, at every level, systems not only contain parts from the lower levels (‘subsystems’) but are also themselves embedded as parts in higher-order systems (‘supersystems’). Nested boxes might illustrate this better than the stacked boxes in Fig. 46.3. Arthur Koestler coined the terms ‘holon’ and ‘holarchy’ to denote systems considered in this way [55].

This is an important systems idea. A core concept of systems thinking is that things are not merely present in environments but are systemically connected to their environments, so everything short of the universe is a part in at least one supersystem. In this context, it can be seen that, in accordance with SP1, it must be that case that system properties are not only emergent over the properties of the parts, but they are themselves subject to submergence as a result of their contribution to their supersystemic context. This implies that systemic properties are determined as a balancing act between the bottom-up influence of the parts and the outside-in influence of the systemic context. This provides a second systems principle (SP2), which I call the ‘principle of universal interdependence’, which I paraphrase as ‘system properties represent a balance between bottom-up emergence and outside-in submergence’.

Note that this principle represents a different idea from the claim often made for systems that they integrate ‘top-down’ causation with ‘bottom-up’ causation—that claim is about system properties acting top-down back on the parts (e.g., the claim that mental properties emerge from brain complexity but then once they exist can act back on the body as in biofeedback). This latter claim is not one about the influence from the systemic environment on a system but is only a more complex view of the workings within the boundary of the system.

The principle of universal interdependence has important implications for science because it means that to characterise a system’s real potential one has to find out not only what the parts contributed (bottom-up causation) but also what was deducted by the supersystemic context (outside-in causation). It means the explanatory arrows go both ways, both down and up from the system boundary. Philosophically, this replaces classical ‘downward only’ reductionism with a kind of holistic interdependence perspective. For scientific research, this then entails that for any phenomenon, the explanatory burden is expanded to now include both bottom-up and outside-in influences and to do so in a balancing way. This principle also has significance for planning interventions and system designs because it implies that there are two interconnected kinds of leverage points for changing system capacity/behaviour, namely via modulation of either the bottom-up or the outside-in influences. This GST principle makes a novel contribution to epistemology by adding a new theoretical virtue: theories and designs will be ‘better’ if they are (more) holistic. This in turn enables us to predict that all the specialised disciplines will become more holistic as they mature (as is already happening in cosmology, biology and medicine). It is therefore likely that a future SE will not only be holistic itself but also increasingly be able to draw on holistic sciences for support.

46.8 Emergence and Complexity

The first two systems principles were derived by looking at interactions without taking complexity into account to a significant extent. However, it is interesting to consider interactions involving diverse kinds of relationships between diverse kinds of parts in the light of the two systems principles derived above.

Consider a supersystem (W) composed of two subsystems, one of high complexity (S1) and one of low complexity (S2). Interactions bind S1 and S2 into a supersystem (W) with emergent new properties, and by SP1 (the conservation of properties) both S1 and S2 must undergo some degree of submergence. The binding interaction that links the two subsystems together is the same for each, but the relative impacts are unequal. For example, consider the impact of gravitational attraction between a very small body and a large one, such as a meteoroid passing a planet. They form a system and each falls towards the other in accordance with

Newton's law of gravity, but the effect on each is very different: the meteoroid's behaviour is strongly conditioned by the nearby planet, but the planet is hardly affected.

Given that the interaction force is the same for each, they must give up the same amount of (gravitational potential) energy, so they contribute equally to the emergence of the new whole. In terms of SP1, they each pay the same amount towards the emergent property of the whole, but the complex subsystem can afford that payment more easily so is less affected by it. In a simple subsystem, the few parts, each has to give up large amount of their energy to make up their contribution to the total, but in the complex subsystem, the many parts, each give up a relatively small amount to make up their contribution. In line with the energy conservation aspect of SP1, this conclusion can be generalised by saying that in systemizing interactions, complex parts pay *proportionately* less towards emergent properties of the whole than simpler parts do. More specifically, it can be said that the impact of submergence on a part is proportional to the complexity differential between the part and the whole. This provides a new systems principle (SP3), which I call the 'principle of complexity dominance', and paraphrase as 'complexity buffers autonomy'.

This principle is significant for scientific research, implying that when characterising the nature and potential of a given system, the two explanatory arrows differ in weight in proportion to the relative complexity of the target system compared with the other systems it is systemically interlinked with. This is an important consideration in the study of naturalistic systems because they cannot be completely shielded from systemizing interactions. This applies equally to the behaviour and performance of designed systems. SP3 is also relevant for planning systemic interventions because the two interconnected leverage points for modulating system behaviour are unequally weighted if there are complexity differentials involved. One corollary of this is that to efficiently control a target system, a more complex one is needed, as suggested in Ashby's so-called 'Law of Requisite Variety'.

46.9 Conclusions and Prospects

In this chapter, I argued that systems principles can be discovered by applying standard scientific and philosophical models to systems concepts, and that this discovery process can take us from basic aphorisms about systems to scientific claims expressed in exact concepts and quantifiable propositions. This can provide useful insight into the nature of systems, and these insights can be valuable for science, engineering, management and philosophy. If this program can be followed through, it would be productive for refining and discovering scientific systems principles that could be used to discover scientifically profound systems laws, which in turn could aid the development of scientifically important systems theories. In this way, a scientifically profound systemology could arise.

The significance of such a development for systems engineering and for society can hardly be overestimated. For example, without a scientific understanding of the nature of fire, it was nevertheless possible to heuristically develop sophisticated applications yielding important products such as ceramic utensils, smoke-cured meats, steel weapons, underfloor heating, lighthouses etc. All these applications depended on the immediate phenomenology of fire. However, once the chemical and electromagnetic nature of fire was understood, the application space was transformed in ways that could not be extrapolated from the basic phenomenology: television, mobile phones, microwave ovens, heart pacemakers, lasers, radio telescopes, and much more. Systemology has the potential to facilitate an equally profound shift via a theoretically grounded systems engineering.

The work presented here is a limited and simple start, and much remains to be done to refine and formalise the models and principles presented here, to incorporate a broader account of worldviews and notions of causation, emergence, lawhood etc., to consider boundary and transitional cases, to distil further systems principles, and to apply them in foundational research in science, philosophy and engineering. However the prospects for such work are possibly now better than they have ever been.

Specific next steps have been identified and some of them are already underway as further projects. The present chapter represents one outcome of a broader research program aimed at developing a general systems framework for research, innovation and design [56, 57]. That broader program has identified six strategies for developing general scientific systems principles (GSSPs) [57], of which the present chapter demonstrates one (namely, deriving GSSPs by applying scientific concepts and principles to general systems ideas) as a pilot project. A project to explore a second approach (leveraging developments in specialised systems disciplines) is underway in the ISSS and will be a major feature of the 2018 ISSS conference. Further projects are now in plan to (a) refine and formalise the principles presented here, (b) expand the network of SSPs by running pilot and refining projects on the other defined strategies, (c) use progress with SSPs to discover scientific general systems laws, (d) integrate these principles and laws into a scientific foundational theory of the nature of complex systems and (e) develop methodologies for applying this theory in scientific research, technological innovation and SE architecting and design. To facilitate collaboration and dissemination, two book projects are under way, one to outline the findings and prospects of the groundwork underlying the present chapter [35], and the other to outline the first version of the general systems research framework entailed by these principles, laws and models.

Acknowledgments I would like to thank Julie Billingham, James Martin, Jennifer Wilby and Duane Hybertson for support and advice in developing the arguments presented in this chapter. Financial and material support for the project was provided by the Centre for Systems Philosophy and by the University of Hull's Centre for Systems Studies. I am also grateful to the three anonymous peer-reviewers for their helpful comments.

References

1. INCOSE (2014) A world in motion – systems engineering vision 2025. INCOSE, San Diego
2. Schindel B (2016) Got phenomena? Science-based disciplines for emerging systems challenges. INCOSE International Symposium 26(1):2256–2271
3. Collopy PD, Mesmer BL (2014) Report on the Science of Systems Engineering Workshop. 53rd AIAA Aerospace Sciences Meeting, American Institute of Aeronautics and Astronautics, 1–4
4. Pennock MJ, Wade JP (2015) The top 10 illusions of systems engineering: a research agenda. Procedia Comput Sci 44:147–154
5. Mobus GE, Kalton MC (2014) Principles of systems science, 2015th edn. Springer, New York
6. Sillitto H (2014) Architecting systems. Concepts, principles and practice. College Publications, London
7. Hitchins DK (1992) Putting systems to work. John Wiley & Sons, Chichester
8. Skyttner L (2006) General systems theory: problems, perspectives, practice, 2nd edn. World Scientific, Hackensack
9. Friendshuh L, Troncale LR (2012) Identifying fundamental systems processes for a general theory of systems. In: Proceedings of the 56th Annual conference, international society for the systems sciences (ISSS). San Jose State Univ, San Jose, p 23
10. Troncale LR (1978) Linkage propositions between fifty principal systems concepts. In: Klir GJ (ed) Applied general systems research. Plenum Press, New York, pp 29–52
11. McNamara C, Troncale LR (2012) SPT II: How to find & map linkage propositions for a general theory of systems from the natural sciences literature. In: Proceedings of the 56th annual conference of the International Society for the Systems Sciences (ISSS). San Jose State Univ, San Jose, p 17
12. Brown MT (2004) A picture is worth a thousand words: energy systems language and simulation. Ecol Model 178(1):83–100
13. von Bertalanffy L (1956) General system theory. Gen Syst 1:1–10
14. von Bertalanffy L (1976) General system theory, Revised edn. Braziller, New York
15. Drack M, Schwarz G (2010) Recent developments in general system theory. Syst Res Behav Sci 27(6):601–610
16. Rousseau D, Billingham J, Wilby JM, Blachfellner S (2016) The synergy between general systems theory and the general systems worldview. Systema 4(1):61–75
17. Troncale LR (1984) What would a general systems theory look like if I bumped into it? Gen Syst Bull 14(3):7–10
18. Francois C (2006) Transdisciplinary unified theory. Syst Res Behav Sci 23(5):617–624
19. Pouvreau D (2014) On the history of Ludwig von Bertalanffy's "general systemology", and on its relationship to cybernetics - part II. Int J GenSyst 43(2):172–245
20. Rousseau D, Wilby JM, Billingham J, Blachfellner S (2016) Manifesto for general systems transdisciplinarity. Systema 4(1):4–14
21. Wiener N (1961) Cybernetics:control and communication in the animal and the machine. MIT Press, Cambridge
22. Simon HA (1962) The architecture of complexity. Proc Am Philos Soc 106(6):467–482
23. Salthe SN (2012) Hierarchical structures. Axiomathes 22(3):355–383
24. von Bertalanffy L (1969) General system theory: foundations, development, applications. Braziller, New York
25. Flood RL, Robinson SA (1989) Whatever happened to general systems theory? In: Flood RL, Jackson MC, Keys P (eds) Systems prospects. Plenum, New York, pp 61–66
26. Francois C (ed) (2004) International encyclopedia of systems and cybernetics. Saur Verlag, Munich
27. Troncale LR (1988) The systems sciences: what are they? Are they one, or many? Eur J Oper Res 37(1):8–33
28. Mulej M et al (2004) How to restore Bertalanffian systems thinking. Kybernetes 33(1):48–61

29. Dubrovsky V (2004) Toward system principles. Syst Res Behav Sci 21(2):109–122
30. Lin F, Cheng TCE (1999) The principles and laws of general systems and their applications. Kybernetes 28(1):75–85
31. Rousseau D, Wilby JM, Billingham J, Blachfellner S (2016) A typology for the systems field. Systema 4(1):15–47
32. Rousseau D, Wilby JM, Billingham J, Blachfellner S (2016) The scope and range of general systems transdisciplinarity. Systema 4(1):48–60
33. Rousseau D, Billingham J, Wilby JM, Blachfellner S (2016) In search of general systems theory. Systema 4(1):76–92
34. Rousseau D, Blachfellner S, Billingham J, Wilby JM (2016) A research agenda for general systems transdisciplinarity. Systema 4(1):93–103
35. Rousseau D, Wilby JM, Billingham J, Blachfellner S (2017) General systemology - transdisciplinarity for discovery, insight, and innovation. Springer Japan, Kyoto. (forthcoming)
36. Dilworth C (1996) The metaphysics of science: an account of modern science in terms of principles, laws and theories. Springer, Dordrecht
37. Psillos S (1999) Scientific realism: how science tracks truth. Routledge, London
38. Schrenk M (2016) Metaphysics of science: a systematic and historical introduction. Routledge, New York
39. Galileo G (1914) Dialogues concerning two new sciences, 1638. Macmillan, New York
40. Rosen R (2005) Life itself. Columbia University Press, New York
41. Gauch HG (2012) Scientific method in brief. Cambridge University Press, New York
42. Kutach D (2014) Causation. Polity Press, Cambridge
43. Bunge M (2000) Energy: between physics and metaphysics. Sci Educ 9(5):459–463
44. Matthewson J, Weisberg M (2009) The structure of tradeoffs in model building. Synthese 170 (1):169–190
45. Maxwell N (2004) Non-empirical requirements scientific theories must satisfy: simplicity, unification, explanation, beauty. [Preprint] URL: vhttp://philsci-archive.pitt.edu/id/eprint/1759. Accessed18 March 2012
46. Rousseau D (2011) Minds, Souls and Nature: A Systems-Philosophical analysis of the Mind-Body Relationship in the light of Near-Death Experiences. PhD Thesis, University of Wales Trinity Saint David, Lampeter UK
47. Bunge M (2003) Emergence and convergence. University of Toronto Press, Toronto
48. Rapoport A (1968) In: Sills DL (ed) General system theory, Int Ency of Soc Sci, vol 15. Macmillan, New York, pp 452–458
49. Schöttl F, Lindemann U (2015) Quantifying the complexity of socio-technical systems–a generic, interdisciplinary approach. Procedia Computer Science 44:1–10
50. Arnold RD, Wade JP (2015) A definition of systems thinking: a systems approach. Procedia Comput Sci 44:669–678
51. Weinberg S (1987) Newtonianism, reductionism and the art of congressional testimony. Nature 330:433–437
52. Koons RC, Bealer G (eds) (2010) The waning of materialism. Oxford University Press, Oxford
53. Gillett C, Loewer B (eds) (2001) Physicalism and its discontents. Cambridge University Press, New York
54. De Caro M, Macarthur D (2010) Naturalism and normativity. Columbia University Press, New York
55. Koestler A (1967) The ghost in the machine. IL Henry Regnery Co., Chicago
56. Rousseau D (2017) A systems philosophy framework and methodology for discovering and leveraging Scientific Systems principles. Presented at the NSF Engineering and System Design (ESD) and Systems Science (SYS) Programs Workshop 2017: Future Directions in Engineering Design and Systems Engineering., Georgia Institute of Technology, Atlanta, 20–22 Jan 2017
57. Rousseau D (2017) Systems research and the quest for scientific systems principles. Systems 5 (2):25
58. Troncale LR (2006) Towards a science of systems. Syst Res 23(3):301–321

Chapter 47
Systems Engineering Pathology: Leveraging Science to Characterize Dysfunction

Heidi L. Davidz

Abstract The formalized methods and enhanced integration of a model-based approach to systems engineering (SE) uncover deficiencies that are more easily masked in ambiguous natural language descriptions and loosely connected artifacts. Model-based systems engineering (MBSE) quickly identifies inconsistencies and process breakdowns. The enhanced precision of MBSE calls for more precision in SE execution as well. In this paper, the systems science concept of systems pathology is extended to the system that executes the SE process. This SE pathology is a methodical way to characterize dysfunction in the execution of the SE process. Through a literature review, systems science and systems pathology are discussed. From an empirical perspective, a framework for SE pathology is proposed, and typical dysfunctions are described. Validation and next steps are discussed. As medicine has been gradually accumulating an understanding of causes, detection, and treatment of diseased states, perhaps SE can begin to accumulate an understanding of causes, detection, and treatment of dysfunctional execution. The objective would be to achieve affordable, healthy SE functions enabled to positively influence program outcomes. SE pathology can provide the holistic approach needed to better characterize dysfunction, for a better chance at intervention, prevention, and ultimately program success.

Keywords Systems pathology • Systems engineering pathology • Systems engineering execution

47.1 Introduction

Moving to a model-based approach for systems engineering (SE) exposes gaps in SE execution. Formalized methods and enhanced integration uncover deficiencies that are more easily masked in ambiguous natural language descriptions and loosely connected artifacts. Although the roots of model-based systems engineering (MBSE) return-on-investment (ROI) are in "automated consistency among work

H.L. Davidz (✉)
External Release Log Number: 045-16, Aerojet Rocketdyne, Rancho Cordova, CA, USA
e-mail: Heidi.Davidz@rocket.com

products when change occurs" and "rapid, definitive answers to stakeholders' questions" [1], MBSE also more quickly identifies inconsistencies and process breakdowns. As MBSE savings are tallied, it is harder to log a savings for a program if a process step now being precisely executed was casually implemented or skipped previously. Certainly, the SE-ROI literature can be employed to justify the more properly executed process step, but in general, the implementation of MBSE calls for a more critical examination of implementation of SE.

The SE literature is full of examples of innovative and useful methods, tools, and processes to execute SE well. However, fewer articles address SE gone wrong. Critical and open examination of SE execution deficiencies enables improvement. The US government Accountability Office issued a report in 2008 [2] on "Defense Acquisitions: Assessments of Selected Weapon Programs." As Phoenix Integration summarized [3], the report found, "an average schedule delay of 21 months and average budget overrun of 26 percent. In dollar terms, the combined cost overrun of all studied programs was $295 billion, up from $42 billion for a similar study conducted just 7 years earlier." That is a tremendous statement. Certainly, the SE function is not fully responsible; however, could better SE execution help?

There is a body of work in "systems pathology," which can arguably be extended to "SE pathology." If one can categorize, characterize, and diagnose SE pathosis, perhaps there is a better chance at intervention and prevention.

47.2 Literature Review

47.2.1 Systems Science

The International Council on Systems Engineering (INCOSE) Systems Science Working Group (SSWG) promotes "the advancement and understanding of Systems Science and its application of Systems Theories to SE" [4]. This is a joint activity of INCOSE and the International Society for the Systems Sciences. This group encourage advancement of systems science principles and concepts as they apply to SE, promote awareness of systems science as a foundation for SE, and highlight linkages between systems science theories and empirical practices of SE.

The group has defined a "Systems Praxis Framework," which is a common framework to bring together concepts from various communities involved in systems work. Their approach is "to create a common language which relates core concepts, principles, and paradigms, allowing specialists from different disciplines to work together more effectively. This vision was captured in a Systems Praxis Framework, a diagram that provided a neutral 'map' which each community can use to explain its own narrative, worldview, and belief system, as well as to appreciate how the various worldviews and belief systems complement and reinforce each other within systems praxis." The details of development are provided in [5], and the result is shown as Fig. 47.1. Note that "systems pathology" is shown in Fig. 47.1 as a theory in "integrative systems science."

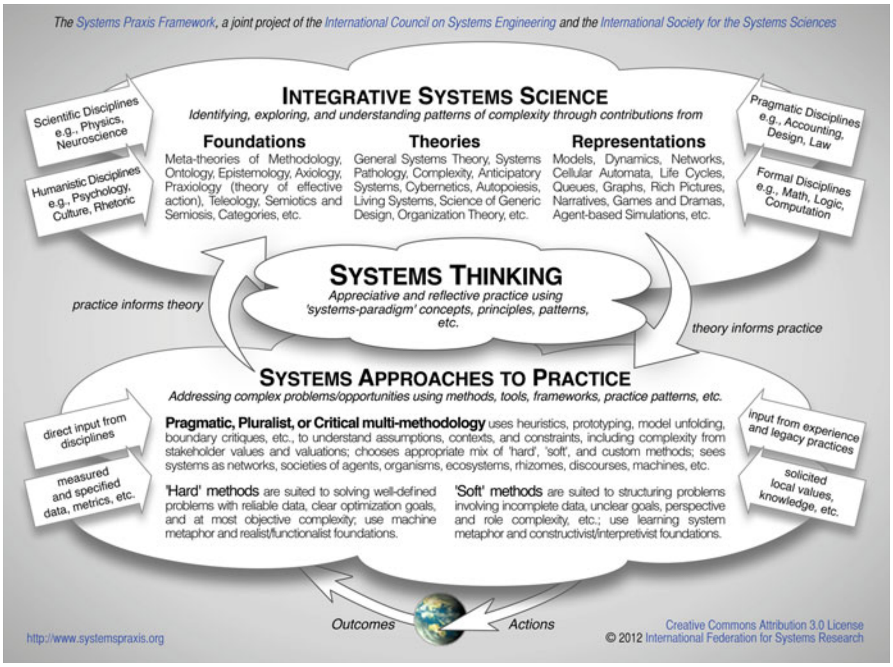

The Systems Praxis Framework, a joint project of the International Council on Systems Engineering and the International Society for the Systems Sciences
INTEGRATIVE SYSTEMS SCIENCE
Identifying, exploring, and understanding patterns of complexity through contributions from
Foundations
Meta-theories of Methodology, Ontology, Epistemology, Axiology, Praxiology (theory of effective action), Teleology, Semiotics and Semiosis, Categories, etc.
Theories
General Systems Theory, Systems Pathology, Complexity, Anticipatory Systems, Cybernetics, Autopoiesis, Living Systems, Science of Generic Design, Organization Theory, etc.
Representations
Models, Dynamics, Networks, Cellular Automata, Life Cycles, Queues, Graphs, Rich Pictures, Narratives, Games and Dramas, Agent-based Simulations, etc.
Scientific Disciplines e.g., Physics, Neuroscience
Humanistic Disciplines e.g., Psychology, Culture, Rhetoric
Pragmatic Disciplines e.g., Accounting, Design, Law
Formal Disciplines e.g., Math, Logic, Computation
SYSTEMS THINKING
Appreciative and reflective practice using 'systems-paradigm' concepts, principles, patterns, etc.
practice informs theory
theory informs practice
SYSTEMS APPROACHES TO PRACTICE
Addressing complex problems/opportunities using methods, tools, frameworks, practice patterns, etc.
Pragmatic, Pluralist, or Critical multi-methodology uses heuristics, prototyping, model unfolding, boundary critiques, etc., to understand assumptions, contexts, and constraints, including complexity from stakeholder values and valuations; chooses appropriate mix of 'hard', 'soft', and custom methods; sees systems as networks, societies of agents, organisms, ecosystems, rhizomes, discourses, machines, etc.
'Hard' methods are suited to solving well-defined problems with reliable data, clear optimization goals, and at most objective complexity; use machine metaphor and realist/functionalist foundations.
'Soft' methods are suited to structuring problems involving incomplete data, unclear goals, perspective and role complexity, etc.; use learning system metaphor and constructivist/interpretivist foundations.
direct input from disciplines
measured and specified data, metrics, etc.
input from experience and legacy practices
solicited local values, knowledge, etc.
Outcomes
Actions
http://www.systemspraxis.org
Creative Commons Attribution 3.0 License
© 2012 International Federation for Systems Research

Fig. 47.1 The systems praxis framework [5]

Chemical engineers take core courses in chemistry and material engineers take core courses in material science. However, systems engineers do not take core courses in systems science. Trained as a biologist, Len Troncale suggests that the field of systems science is rapidly fragmenting, and many partial alternative approaches exist such that a consensus systems science core course has failed to emerge [6]. Within the SSWG, Troncale leads the "Systems Processes and Pathologies" project, which has two parts. First, the systems processes project identified the 100+ systems processes found in natural systems. These are then being categorized and characterized. As stated in [4], "By looking at natural systems and seeing how they address their situation and continue to thrive in their environment under adverse conditions, this will give us insights into how we can better develop man-made systems." Second, the systems pathologies project is examining existing systems and their failure modes to identify a generalized set of system dysfunctions. Once these are identified and characterized, the group will look at how natural systems have evolved to overcome these pathologies. "These disease avoidance or recovery mechanisms will be cataloged to help the systems engineer design a system that is more tolerant of faults and failures."

The first project on systems processes is not fully complete, as it is a tremendous task to integrate the plethora of systems theories, sources, approaches, and tools to enable the unified science of systems as a fundamental basis for SE. The synthesis to date is explained in [6, 7]. This includes sources for systems science synthesis, the protocol for integrating systems literatures, the guiding tenets for the project of unifying systems sources, and the synthesis to date. Example fields, classes, and theories that are sources for unification of systems science include general systems theory (GST), operations research, systems dynamics, systems biology, agent-based modeling, systems neurobiology, etc. The current working list of 55 key systems processes are also shown in this reference. Examples include adaptation processes, binding processes, emergence processes, feedback (general), growth patterns and laws, input processes, redundancy processes, etc.

Convergence of systems science enables evolution in MBSE. One example is the INCOSE MBSE Patterns Working Group whose purpose is "to advance the availability and awareness of practices and resources associated with the impactful creation, application, and continuous improvement of MBSE Patterns over multiple system life cycles" [8]. As systems science codifies patterns, they can be applied more systematically. The INCOSE Ontology Working Group proposes "the application of ontology based knowledge management as an orthogonal driver within SE practice to enable processes, methods and tools harmonization" [9]. One project for this group is MBSE interoperability through a common knowledge management paradigm. As systems science develops, this informs evolved application of MBSE.

47.2.2 Systems Pathology

Some in the SSWG argue that these 55 key systems processes do not just describe how systems work, but this could lead to better understanding of how systems do not work. As Troncale states [6], "Each of the key systems processes could be examined in case studies for not achieving the function they normally perform in making a system sustainable. That quickly would yield a 'taxonomy' or 'classification' of possible dysfunctions that is much more detailed than currently possible. Each systems process would then name an entire category of dysfunctions for SEs to be on the lookout for or avoid by design." He proposes to mimic the approach of medicine to disease and adapting that to the design and maintenance of systems.

Here are some of the categories of system dysfunction Troncale has identified [6].

- Cyberpathologies (dysfunctions of feedbacks)
- Rheopathologies (dysfunctions of systems flows)
- Cyclopathologies (dysfunctions of cycling, oscillations)
- Heteropathologies (dysfunctions of hierarchy or modular structure)
- Hapsopathologies (dysfunctions of network structure or dynamics)
- Teratopathologies (dysfunctions in developmental processes)
- Stathopathologies (dysfunctions in stability states)

Expanded examples of several of these dysfunctions are provided in [6]. As an illustration, here is the expanded example for cyberpathologies, "Cyberpathologies (abnormal delays of feedback loops relative to response needed; mismatch in increments or degree of change with needed magnitudes; mistake in or absence of coupling of negative and positive feedbacks; dysfunction due to feedback not present at all; missing feedback across hierarchical levels; feedback connect to wrong part of interacting net; dysfunctional change in output no longer calibrated to need in systems environment)."

Once each of the dysfunctions has been recognized, it is suggested in [6] that, "this new Systems Pathology would provide a ready framework for studying each specific disease as medicine has done for 2000 years gradually accumulating an understanding of causes (etiology), a better way of detecting each dysfunction (diagnosis), and a tighter coupling of alternative treatments with outcomes (prognosis). The increased knowledge of how systems don't work could help avoid problems in the earliest stages of describing the requirements of a needed system, and in the design and maintenance of a system."

47.2.3 Pathologies in Systems of Innovation

In connection with the systems processes and systems pathologies project of the INCOSE SSWG, Bruce Beihoff and William Schindel describe "systems of innovation" (SOI) and the characteristics of health and pathology in these systems [10]. For living ecosystems, innovation adapts to changes by predators, prey, and environment. For engineered systems, innovation exploits market interests in new capabilities; creates new markets; develops competitive advantage; and adapts to changes in technologies, infrastructure, regulations, and environment. In this work, the process of innovation itself is described as a system, the SOI.

In this context, the definition of system health is if the system "performs (externally and internally) in a manner typical of other systems of the same type in like external circumstances." The authors suggest systems pathology is then when the system fails "to perform (externally or internally) in the manner typical of other systems of the same type in like external circumstances." It is interesting to note that in these definitions, performance is required. A system is not deemed healthy just because it has the capability to perform well, which is a key distinction. An initial SOI pathologies catalogue is provided.

47.2.4 Systems Pathology in Medicine

In the medical community, there is movement toward a more systems-level approach to the traditional field of pathology. In this context, one systems pathology definition is [11], "the study of disease through the integration of clinical, morphological, quantitative, and molecular parameters using mathematical analytical frameworks. The aim is to create coherent models which enable the understanding of pathophysiological processes in their entirety and generate hypotheses that can be tested experimentally." As noted in [12], "the wealth of morphological, histological, and molecular data from human cancers available to pathologists means that pathology is poised to become a truly quantitative systems science."

Jose Costa notes how technological advances and dramatic increase in computational power have injected new life into systems-level thinking in medicine [13]. He emphasizes the close relationship of systems pathology to systems biology. He notes that, "The idea of system as comprised of interdependent elements has been the subject of human inquiry for millennia. The realization that an ensemble of numerous interconnected elements influence and cause events in the world that surrounds us is recorded in the writings of the pre-Socratic school of thought which laid much of the foundation for the natural sciences." Considering that much of what is studied in physiology and pathology deals with phenomena occurring in complex organisms, a means for system thinking has become essential for the elaboration and integration of scientific biomedical knowledge. He notes that the foundation of this necessary systems thinking is GST. Costa says that GST is a method that can be used to gain predictive insight into how life works in both normalcy and disease. The translation from systems biology to systems pathology has been relatively easy and natural. "As it has always been the case, disease has also served as a good model for physiology and much about normalcy has been learned from disturbed systems that constitute diseased states." He also notes that, "When it comes to solving a clinical problem, the master clinicians of the 19th and 20th century relied on their observational power and their ability to recognize patterns that in their experience were repetitive." However, as biomedical sciences have made inroads in the understanding of disease at the molecular level, advanced classifications are possible. He notes agreement that any tension between the classical clinical approach and new technologies is resolved by integration and collaboration.

47.2.5 Terminology

Systems engineers may not be familiar with medical terminology, so here are some clarifications on definitions. The term pathology may be used broadly to refer to the study of disease in general or more narrowly to describe work within the medical field of pathology. In medicine, it is the branch of medicine concerned with the study of the nature of disease and its causes, processes, development, and consequences [14]. The study of disease in general includes plant pathology and veterinary pathology.

Note that the word "pathology" is sometimes used as a synonym of disease or pathosis. However, others insist on differentiating pathosis and pathology, where pathosis is the deviation from a healthy function and pathology is the study of that deviation from healthy function [14]. Some style guides avoid the use of the word to describe illness as the "-ology" form is reserved for the study of disease. Pathosis is the term used for "any deviation from a healthy or normal structure or function; abnormality; illness or malformation." Natural usage perpetuates use of the term pathology as disease instead of the study of disease. "Another limitation is that pathology meaning 'illness' has an adjectival form (pathologic), but the corresponding adjectival form of pathosis (pathotic) is idiomatically missing from English." Pathologic is then used for both "diseased" and "related to the study of disease." This keeps the illness sense of pathology in natural use.

Related definitions include etiology as the cause, set of causes, or manner of causation of a disease or condition. Diagnosis is the identification of the nature of an illness or other problem by examination of the symptoms. Prognosis is the likely course of a disease or ailment. Symptom is a subjective indication of a disorder or disease. As MBSE ontology evolves, perhaps some of the pathology terminology could be included.

47.3 Proposed Framework

47.3.1 SE Pathology

As intended by the INCOSE SSWG and Troncale, "systems pathology" is used to understand how the key systems processes dysfunction, to detect or avoid this dysfunction by design. If the system of interest extends to the sociotechnical system that performs SE, "systems pathology" could extend to "SE pathology." The system of interest moves from the "system being designed" to the "system that designs the system being designed." If the principles of systems theory are applied, it does not matter if the system of interest is the system being designed or the system that designs the system being designed.

Extrapolating from the work of Costa [13] who said that "disease has also served as a good model for physiology and much about normalcy has been learned from disturbed systems that constitute diseased states," SE pathology can examine how SE works in both normalcy and disease and much about SE normalcy can be learned from disturbed systems that constitute diseased states.

In a professional society technical meeting, an individual once noted that, "A 'good' systems engineer can overcome a bad environment. If you know it's going to rain, you bring an umbrella. If you have systems thinking and strong systems engineering competency, you can overcome bad systems engineering execution." However, the sector, enterprise, business, or even team dynamics may not be such that an individual systems engineer (or SE group or company) can have an impact significant enough to correct SE execution.

There are patterns by which SE execution fails. More formally characterizing these patterns of dysfunction (diagnosis) could lead to accumulated understanding of causes (etiology), prediction of outcomes (prognosis), and tighter coupling of alternative treatments. The author is particularly interested in this last opportunity – collected strategies and tactics to overcome specific dysfunctions. Knowledge of how SE execution does not work can also help avoid problems when the SE process can be intentionally designed.

If, as Troncale states, medicine has for 2000 years been gradually accumulating an understanding of causes, detection, and treatment of diseased states, perhaps SE can begin to accumulate an understanding of causes, detection, and treatment of dysfunctional SE execution. The objective would be to achieve affordable, healthy SE functions enabled to positively influence program success. SE execution problems have sometimes been met with corrective mechanisms such as providing SE training and improving SE process; however, isolated interventions are often not enough. SE pathology can provide the holistic approach needed to better characterize dysfunction, for a better chance at intervention and prevention.

Similar to the current suggested application of systems science to MBSE patterns and MBSE ontology, systems pathology can also be applied to MBSE patterns and MBSE ontology. Systems pathology extended to SE pathology can identify patterns of dysfunction and a common knowledge management paradigm for dysfunction.

The failure of SE efforts might not be evident for months or years after the SE work is completed. This timeframe to detect failure may be shorter for other engineering disciplines. For example, a mechanical engineer may find an error in design during simulation or initial component testing. As the result of poor SE execution may not be found for a long time compared with other types of engineering, identifying patterns of SE dysfunction prior to failure becomes even more important.

47.3.2 Framework for SE Pathology

Costa noted that the master medical clinicians of the nineteenth and twentieth centuries relied on their observational power and their ability to recognize patterns that in their experience were repetitive, then technological advances and increased computational power enabled more advanced understanding of disease [13]. Similarly, master systems engineers in the twentieth century have relied on observational power and their ability to recognize that patterns in their experience were repetitive. However, in light of technological advances and systems science, perhaps systems engineers can now better understand, diagnose, and treat SE dysfunction.

Building on the concept of systems science leading to systems pathology, which can be extended to SE pathology, Table 47.1 provides a proposed framework for SE pathology. In this use, the word pathology is being used as the “study of” and the word dysfunction is being used for the pathosis. As an initial hypothesis from which to work, the table is conjecture based on case studies, empirical observation during

Table 47.1 Proposed framework for SE pathology

	Dysfunction	Description
1.	Oversized	SE effort too large for the program size, violating the SE-ROI guidelines
2.	Undersized	SE effort too small for the program size, violating the SE-ROI guidelines
3.	Façade	Illusion of SE in place, but close examination of artifacts show SE activities not performed
4.	Timeliness	SE artifacts not produced in time for design impact, perhaps generated only for contract deliverable
5.	Disregard	SE process not perceived to be of value in the cultural context
6.	Unskilled	Inadequate knowledge, skills, and abilities in SE for those performing tasks
7.	Quality	SE effort performed is of poor quality
8.	Pace	Program pace is too quick, not allowing enough time to perform adequate SE
9.	Confusion	SE activities not coordinated
10.	Panic	Execution driven by the latest emergency, not the intended process

the author's dissertation research, literature on SE execution, and personal experience at several companies [15–18]. This list provides a starting point for later validation using a more rigorous and systematic approach. The dysfunctions listed here are process issues; however, they can result in both process and product failures. The link to the systems pathology literature is use of the general idea of SE pathology. Section 48.3.3 discusses how theoretical systems pathology could be linked to an empirical framework for SE pathology.

To validate the framework, a case study approach could be used, where SE execution failures are assessed, characterized, and grouped. Or, a survey of expert systems engineers could be conducted, where they review the framework and suggest revisions. An initial validation of this framework was attempted by referencing 18 years of "SE" peer-reviewed journal articles to address SE pathology and characterize SE dysfunctions. However, a more systematic examination of SE failure in particular is needed. With the sensitivity involved in information related to failures, sampling bias and repeat validity are of particular importance when framing a research design.

Current SE standards and guidance such as the INCOSE Handbook [19] and the Defense Acquisition Guidebook [20] describe proper SE processes. There are SE competency models to describe characteristics of good systems engineers. However, there is opportunity to characterize SE failure in particular. Some of the dysfunctions in Table 47.1 are antifunctions of good processes; however, some of the dysfunctions such as "façade" are arguably not a direct antifunction and would require case study.

As the "SOI" described in [10] are possibly comparable to the systems executing SE, the list of dysfunctions given in the SOI pathology catalogue may be another source for categorization and groupings.

In 2013, Eric Honour published his thesis on, "Systems Engineering Return-on-Investment" (SE-ROI), which provides guidance on SE sizing [21]. He discovered statistically significant relationships between SE activities and three success measures – cost compliance, schedule compliance, and stakeholder overall success. SE-ROI is discovered to be as high as 7:1 for programs with little SE effort and 3.5:1 for median programs. In that work, he determined that the, "optimum SE effort for median programs is 14.4% of total program cost."

Building from these results, the dysfunction of "oversized" is when the SE effort is too large for the program size, violating the SE-ROI guideline of 14.4%. When the SE effort is oversized, the program is at risk of excessive costs for SE tasks.

Similarly, the dysfunction of "undersized" is when the SE effort is too small for the program size, violating the SE-ROI guideline of 14.4%. In this situation, there are not enough resources to execute the needed SE tasks to realize the cost compliance, schedule compliance, and stakeholder overall success enabled by the SE activities. There may also be lack of SE follow through, where good SE was done early in the program, but SE guidance is not used in subsequent life cycle stages.

"Façade" is when an illusion of SE is in place, but a closer examination shows that SE activities were not truly executed. To discover this situation, SE competency is needed as one examines details of the artifacts of the executed process. Cutting through the "smoke and mirrors" requires SE knowledge and examination of details.

"Timeliness" is when SE artifacts are not produced in time for design impact. Perhaps the artifacts are generated for a contract deliverable, but the timing is such that the effort has already missed the window of opportunity to impact the design. As an example, an extensive failure modes and effects analysis might be generated, but too late in the design process to truly impact the design. SE may be done too early in the program before information is available. Pairing of appropriate SE tooling to the program stage is also important.

"Disregard" is when the SE process is not perceived to be of value in the cultural context. It cannot be assumed that SE is perceived as a value-add function. This is a "given" in some organizations but not others.

"Unskilled" is when those performing SE have inadequate knowledge, skills, and abilities to perform the task. This is a dysfunction that is often the focus of corrective actions. The Atlas project is an example of an effort to address this dysfunction [22].

"Quality" is when the SE effort performed is of poor quality. The sizing of the SE function might be correct, and the schedule may be adequate, but the quality of the SE work as executed is poor. Misuse of modeling tools and methods will affect quality. Tooling and techniques must match the problem.

"Pace" is when the program pace is too quick, and there is no time for adequate SE. As opposed to "undersized", there may be adequate resources to perform the SE tasking, but there was no time to execute the tasks. For "timeliness," the SE effort may be completed, but not in time for design impact; where for "Pace," the SE effort might be skipped completely as the program is moving too fast.

"Confusion" is when the SE activities are not coordinated. Schedule and cost may be adversely affected as team members figure out who is doing what and possibly performing duplicate work.

"Panic" is when the execution is driven by the latest emergency, not the intended SE process. In this case, process control may suffer as "fire-fighting" determines task priority.

As follow-on work is performed, the discussion of these dysfunctions can be elaborated, as the discussion will then be based on agreed-to disclosures of SE execution problems. Published case studies in the open literature could be used. However, it would be useful to have deeper details and multiple perspectives to gather a more complete understanding of the dysfunction(s) that occurred.

In the "lean" and continuous improvement literature, a "star part" is tracked through the value stream to trace processing steps and record times. Similarly, an SE artifact could be assigned as the "star part" and one can trace that artifact through the process to see what is actually going on in the evolution of the design. As an example, a pressure requirement could be assigned as the "star part" and the origin and evolution of that requirement could be traced to identify process abnormalities experienced.

Once these patterns of dysfunction are formally identified, they can further evolve the formal application of MBSE methods. Patterns and ontology are again examples. Patterns of dysfunction could be codified. Modes of failure could add to the ontology. At some point, perhaps automated logic checking could scan models to sense potential SE pathosis. Note that SysML is a popular MBSE approach; however, there are opinions that other methods like Business Process Modeling Notation (BPMN) are better for process modeling. It is another area of research to see if BPMN can be used in modeling SE process definition and execution well enough to identify potential pathosis.

47.3.3 Linking Systems Pathology to SE Pathology

The proposed framework for SE pathology could be empirical, from study of SE failures. The systems science key processes lead to systems pathologies, and it appears there may be a link between the "top-down pathologies" from the systems science literature and the "bottom-up" dysfunctions in the proposed framework for SE pathology. For example, "Façade" may link to "Cyberpathologies" as this is an abnormality in feedback loops. "Panic" may link to "Rheopathologies" as this is a disruption in flow. "Disregard" may link to "Hapsopathologies" as this is unstable connection between nodes. "Unskilled" may link to "Teratopathologies" as it is dysfunction of developmental processes. As the unifying systems science and corresponding systems pathology evolve, these top-down systems pathologies could be used to identify "most likely errors" in SE execution far ahead of time. Though, rigorous work would be needed to validate connections.

47.3.4 Using the Framework for SE Pathology

As the study of SE pathology develops, each dysfunction (once validated) may have corresponding symptoms, diagnosis, causes, prognosis, and treatments. Patterns of dysfunction in SE execution could be paired with recommended treatments for actors with varying degrees of influence. Yes, some of the treatments will likely be logical, but many illogical things happen in the interworking of intertwined systems, and clear identification of patterns can be helpful.

Tools such as SE leading indicators [23] certainly inform when SE is off track. However, some organizations are not advanced enough to have these in place. Standard program management health checks such as cost and schedule variances also warn of execution problems. However, for dysfunctions such as “façade,” the milestones and schedule may be met, but the intended technical content is not of necessary maturity. Traditional measures being used in a single organization may not bring sufficient insight to detect SE dysfunction across the supply chain. However, these traditional measures could be “symptoms” to inform the diagnosis, etiology, prognosis, and treatment.

Considering “timeliness” as the dysfunction, the symptom (subjective indication of a disorder or disease) could be a comparison of SE artifact delivery date versus need date to impact design, which indicates the SE artifact is too late in the life cycle to impact the system design. The diagnosis (identification of the nature of an illness or other problem by examination of the symptoms) is “timeliness.” The etiology (cause, set of causes, or manner of causation of a disease or condition) could be customer review iterations and delays. The prognosis (likely course of a disease or ailment) could be excess program costs for deliverables not impacting design. The treatment could be negotiating with customer on “pencils down” completion dates. An example could be design verification planning records intended to be completed prior to verification activities in customer review iterations until after the verification activity is complete and documented, which is too late to drive the verification activity.

47.4 Next Steps

The immediate next step would be to use the framework provided as a foundation for a more systematic evaluation of SE dysfunction. Further conversations with those working in systems science and systems pathology should be undertaken for maturation of an approach for evolving SE pathology.

For practitioners, there is an interest in understanding causes (etiology), detecting dysfunction (diagnosis), and treatments with outcomes (prognosis). For a systems engineer caught in a system experiencing dysfunction, the key nugget of value is tactical strategy to impact treatment. For those in a more powerful position, knowledge of how systems do not work can help avoid problems in the earliest stages of describing the requirements of a needed SE system and in the design and maintenance of that SE system.

Realistically, a typical systems engineer in industry does not have time to think deeply about the theoretical underpinnings of the system science, which drives the dynamics controlling his/her daily experience. However, clear and concise categorization of SE dysfunctions with corresponding recommended treatments could bridge the gap for tactical implementation.

The increased precision and effectiveness of a model-based approach to SE demands a more precise handling of SE execution as well. As the formalized methods and enhanced discipline integration of MBSE uncovers gaps and deficiencies, SE pathology could more rigorously identify dysfunctions in SE execution and corresponding treatment.

47.5 Conclusion

With formalized methods and enhanced integration, a model-based approach to SE quickly identifies inconsistencies and process breakdowns. The enhanced precision of MBSE calls for more precision in SE execution as well. A literature review was presented to summarize systems science and systems pathology. An extension to SE pathology was developed. A framework for SE pathology was proposed, and descriptions of typical SE dysfunctions were given. Methods for validation and further development of this framework were suggested. Next steps were proposed. In the field of medicine, disease has served as a good model for physiology, and much about normalcy has been learned from disturbed systems that constitute diseased states. Similarly, it is argued that much about proper SE execution can be learned from disturbed systems that constitute dysfunctional SE execution. Through a methodical examination of dysfunctional states, SE pathology can provide understanding of how to design systems and effectively intervene for healthy SE execution.

References

1. Delligatti L (2015) “MBSE with SysML: automated consistency with rapid, definitive answers,” webinar presented March 12, 2015
2. United States Government Accountability Office (2008) “Defense acquisitions: assessments of selected weapon programs,” Report to Congressional Committees, GAO-08-467SP, March 2008
3. Ko Andy (2016) Phoenix Integration, Inc., “MBSE Ecosystem: System and Analytical Models Integration for Critical Systems Simulation,” webinar presented on September 8, 2016
4. International Council on Systems Engineering (INCOSE) Systems Science Working Group (SSWG), joint activity of INCOSE and the International Society for the Systems Sciences (ISSS), webpage available at: https://sites.google.com/site/syssciwg/, accessed October 2016

5. Martin James, Bendz J, Chroust G, Hybertson D, Lawson H, Martin R, Sillitto H, Singer J, Singer M, Takaku T (2013) Towards a Common Language for Systems Praxis
6. Troncale L (2013) “Systems processes and pathologies: creating an integrated framework for systems science,” presented at the 2013 International Council on Systems Engineering (INCOSE) Symposium, Philadelphia
7. Troncale L (2013) “Unifying systems theories and systems pathology,” INSIGHT, 16, 1, April 2013, John Wiley & Sons, Inc., Hoboken, NJ
8. INCOSE MBSE Patterns Working Group, “INCOSE MBSE Patterns Working Group Charter”
9. LLorens J, Anabel F (2017) “Ontologies Working Group” presentation to the Systems Science Working Group at the INCOSE International Workshop, 2017
10. Beihoff B, Schindel W (2012) “Systems of Innovation I: Summary Models of SOI Health and Pathologies,” International Council on Systems Engineering (INCOSE) International Symposium, July 2012, Rome, Italy
11. Faratian D, Harrison D (2013) Systems pathology. In: Encyclopedia of systems biology. Springer, New York, pp 2097–2099
12. Faratian D, Clyde R, Crawford J, Harrison D (2009) Systems pathology – taking molecular pathology into a new dimension. Nat Rev Clin Oncol 6(8):455–464
13. Costa J (2012) Systems pathology: a critical review. In: Molecular oncology 6, 1, 27–32, Federation of European Biochemical Societies, published by Elsevier. Netherlands
14. Wikipedia, “Pathology” and “Pathology Usage Notes,” https://www.wikipedia.org/. Accessed Feb 2017
15. Davidz H (2006) Enabling systems thinking to accelerate the development of senior systems engineers,” Ph.D. dissertation, Engineering Systems Division (ESD), Massachusetts Institute of Technology, Cambridge
16. Davidz H, Nightingale D (2008) Enabling systems thinking to accelerate the development of senior systems engineers. In: Systems engineering, 11(1), John Wiley & Sons, Inc. Hoboken, NJ
17. Davidz H, Martin J (2010) Defining a strategy for development of systems capability in the workforce. In: Systems engineering, 14(2), John Wiley & Sons, Inc. Hoboken, NJ.
18. Davidz H (2016) Invited chapter “Insights based on an empirical study of systems thinking development”. In: Frank M, Shaked H, Koral-Kordova S (eds) In the book “Systems thinking: foundation, uses and challenges”. Nova Science Publishers. New York
19. INCOSE (2015) Systems engineering handbook: a guide for system life cycle processes and activities, 4th edn. INCOSE-TP-2003-002-04. INCOSE. San Diego, CA
20. United States Department of Defense, “Defense Acquisition Guidebook,” Chapter 4 Systems Engineering
21. Honour E (2013) “Systems engineering return on investment,” doctoral thesis, Defence and Systems Institute, School of Electrical and Information Engineering, University of South Australia
22. Pyster A, Dominick P, Henry D, Hutchinson N, Lipizzi C, Kamil M, Manchanda S (2014) “Atlas: The Theory of Effective Systems Engineers, Version 0.25,” Systems Engineering Research Center, Technical Report SERC-2014-TR-038-4, November, 2014
23. Roedler G, Rhodes D, Schimmoller H, Jones C (2010) “Systems engineering leading indicators guide,” Version 2.0. INCOSE-TP-2005-001-03. Massachusetts Institute of Technology, INCOSE and PSM. Seattle, WA

Chapter 48
Using the PICARD Theory as a Tool to Improve Systems Thinking Ability

James N. Martin

Abstract Systems are not as real as we think they are. It is indeed true that engineers design things that when built and placed into service are very real. But these deployed things are merely an embodiment of the system's design. We make the common mistake of thinking that the deployed artifact is the "system." The engineering design is a system as a "concept." The elements of this conceptual system are the things involved in systems thinking. We conceive of various elements that come together as systems to examine their contribution to the system's intended "purpose." The PICARD theory was formulated to help the systems engineers and others to more readily see and understand the systemic aspects of a situation and to better employ systems thinking techniques. This paper will explore what it means to employ systems thinking to imagine various system structures and to examine these structures for their suitability in different situations to address current or anticipated problems.

Keywords Systems thinking • Systems theory

48.1 Introduction

We have a common (mis)perception that the systems we engineer are real. They are in fact imaginary. By "real" I mean things that actually exist in the real world. By "imaginary" I mean things that exist in the mind. Nothing is inherently a system. Systems exist only in the mind. There might be parts of the system that are real, but the system as a whole is a fabrication of our mind. Most of us believe that a system exists of its own accord. But we have free choice in choosing the constituents and boundaries of such systems.

Take for example the solar system. It was traditionally defined as the Sun plus nine planets. Then in 2006, a group of astronomers decided that the solar system *really* consists of the Sun plus eight planets. Did Pluto disappear from the sky? Of course not – they merely realized that the previous definition of the system was not

J.N. Martin (✉)
The Aerospace Corporation, El Segundo, CA, USA
e-mail: James.N.Martin@aero.org

A.M. Madni et al. (eds.), *Disciplinary Convergence in Systems Engineering Research*, https://doi.org/10.1007/978-3-319-62217-0_48

suitable to their way of thinking. In the same manner, systems engineers should define their system of interest in a way that is most suitable to their situation. Sometimes the system of interest should be larger than just the thing being acquired to meet a mission need or to solve a particular problem.

This false notion that engineered systems are “real” (i.e., just waiting to be discovered) often leads us to only think about the physical aspects of systems and not their sometimes more important nonphysical attributes such as behavior, information, and value. If we do not change the manner in which we conceptualize and define a system, we will not likely examine other possible constituents or boundaries of the system to determine if this might lead to a better solution.

A system is commonly defined as two or more parts, which interact in a way that produces some property not possessed by the parts themselves. This is a good way to think about systems that are simple, but it is not adequate when dealing with large, complex systems, or those that are not completely physical in nature such as sociotechnical systems. A more enlightened view of systems is needed – which leads us to the PICARD theory that provides a mental framework for thinking about systems in a more holistic manner.

The PICARD model was developed originally as a way to teach people in a systems engineering class how to better think about and deal with systems. Those coming from the regular engineering disciplines were unfamiliar with the notion of a system and how it was different than the usual products they were dealing with. Often they did their engineering in the physical realm and had difficulty in understanding the concept of “function” and the interactions that go on between functions. They could not grasp the difference between the functional architecture and the physical architecture. They had difficulty appreciating why you would want to work in the “function space” to examine potential solutions as they were so used to doing everything directly on the physical objects of their design. The PICARD ideas helped them to expand beyond their more customary focus on the product only.

48.2 The PICARD Theory of Systems

It is by examining these six types of things that true systems thinking can occur. I call this the PICARD theory of systems (products, interaction, context, action, relationship, and destiny). The connections between the PICARD elements of the system are what give the system its “emergent” behavior. Each product within a system will have its own behavior, but this behavior is often of little use by itself until it creates a system-level behavior that provides a useful feature or function to the system user or operator.

System = Products
+ Interactions + Context
+ Actions + Relationships + Destiny

48.3 Background

The car you drive has physical traits like size, weight, and power. But in your mind, you also think of the car as to how it grabs the road, how it feels to drive in traffic, and how it propels you to work every day. The car you *imagine* is not the same as the car that sits in your driveway. Our mind is like that – it makes us believe that the *thing* is identical to our *concept* of the thing [1]. Whenever we think about the car we are really thinking about the car *concept*, not the car *object*. Of course, the car concept has direct connection to the physical attributes of the car object. Let us examine these connections more closely.

48.3.1 Relationships are Key

One way to think about these *connections* is through the discipline of systems thinking [2]. By using this way of thinking, we can better "see" the car as a system, for example, as a collection of things that interact with each other and with the driver and the road. But it is more than just the *interactions* (i.e., mutual or reciprocal actions or influence) that give the car its useful properties. Interaction involves exchange of data or information between the parts, or a transfer of forces or energy across the interfaces. To fully understand the car as a "system," we must also consider the *relationships* between the parts.

The driver "feels" as one with the car, feels the road and its curves, and feels good about driving the car. These feelings are relationships between the driver (as part of the car system) and other parts of the car or things outside. For the car to drive well, there needs to be a very specific spatiotemporal relationship between the four wheels. If this relationship gets goes awry, then the performance could be compromised and the safety of the passengers could be in jeopardy. The wheels do not actually interact with each other, but their relationships to each other are an important feature of the car as a "system."

Our seeing of these relationships[1] is key to what constitutes systems thinking. Without relationships, there can be no system – we cannot even "think" about the

[1]A relationship is an association of some sort between two or more things. The "things" can be thought of as the *nouns* of the world and the relationships can be thought of as the *verbs* of the world. A song (thing) "is performed by" (relationship) a singer (thing). A concert "is performed by" an orchestra. An orchestra "consists of" people. Orchestra members "play" instruments. And so on.

system unless there are relationships to think about. If the parts of a system have no relationships between them, then there can be no system. Our task as a systems engineer is to discover the most relevant relationships and determine how they best contribute to overall value [3]. However, systems engineers are usually taught to only consider the interactions between parts as being important. This is where the PICARD theory can serve as a useful mental model of what a system really consists of so that by having a better grasp of the actual situation, better solutions can likely come about.

48.3.2 What is a System?

We can think of any object (or collection of objects) as a "system." When we think of a single object as a system, this is called "black box" thinking. It is called a black box because we are ignorant of (or do not care) what is inside the box. The focus is on the interaction with things outside the box [4].

When we think of a collection of objects as a system, this is called "white box" thinking as we are looking inside the box to see what is happening. The white box, in this case, is the collection (or container) for the things we consider together as a system. When you are not concerned with all the internal workings of the box, but only certain aspects of it, this is sometimes called "gray box" analysis.

The systems approach to thinking can help us examine how the system behaves, how well it performs, what it is made of, or perhaps if it has value. This examination is often called systems analysis [5]. Before we can do systems analysis, we must define the system in terms of its objects, their relationships, and the context in which these objects reside.

Thinking of something "as" a system is usually not consciously done; we often do this below our level of conscious thought. People who are better at systems thinking seem to have learned to force this sort of thinking more into the conscious area of their mind. They have learned mental tricks to help manipulate their perceptions to consider more of the "systemness" of a situation vice the merely substantive, material aspects of those circumstances.

One technique for facilitating this thinking process is to draw a context diagram. This diagram has a box that represents the system with inputs and outputs going into and out of the system [6]. We can then decompose this system (the black box) into its constituent parts and then examine the relationships (e.g., inputs/outputs) for each of the parts (the white box).

We can go into more detail by examining each part as a system itself by going through the same decomposition process, ad infinitum, until we discover what we need to know about that system. We can recompose these lower level parts into higher level aggregations to discern where we have emergent behavior that is not exhibited by the parts taken individually [7]. Some of this emergent behavior is good and some is bad. We are trying to find the system configuration where the good behavior outweighs the bad behavior in a cost-effective manner.

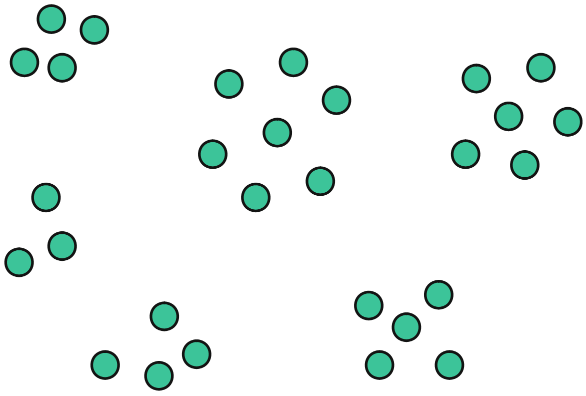
For the purpose of this paper, let us think of a system in this way:

A system is a conceptual overlay placed on top of those things we choose to consider as elements of the system.

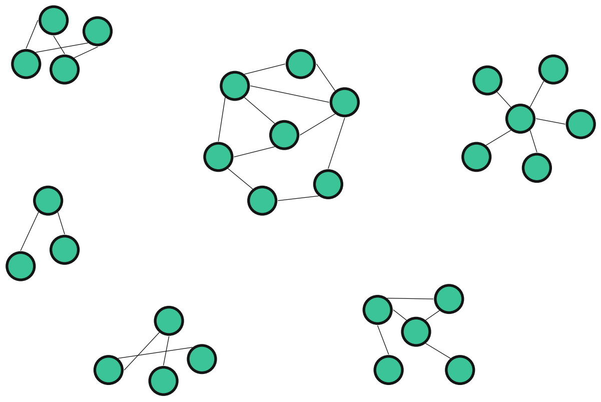
This definition should suffice for now. In Sect. 48.3 of this paper, I will describe the six dimensions of a system that can help us better see every important aspect of the system of interest.

48.3.3 Enabling Systems Analysis

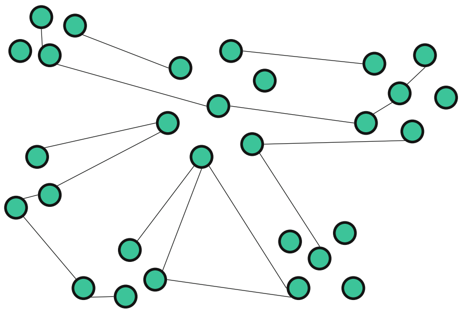
When we need to do analysis of a system, we need to be careful in how we choose the system of interest to be examined. Making a wrong choice can possibly lead to erroneous conclusions. To illustrate this concept of system choice, let us look at a situation where there are various things.



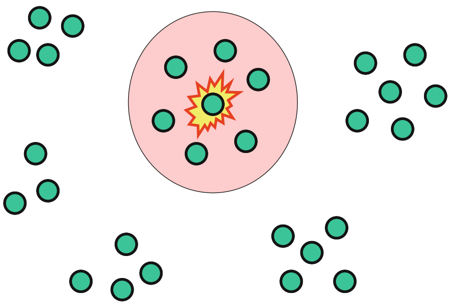
It is quite natural, and perhaps even fitting, that you group these items into several systems like this based on strong interactions due to couplings of some sort – perhaps physical attractions due to gravity or some other physical force, or maybe data flows:



But you can examine these things from another angle, such as mission threads or social relationships perhaps, and your depiction of relationships might appear like this instead:



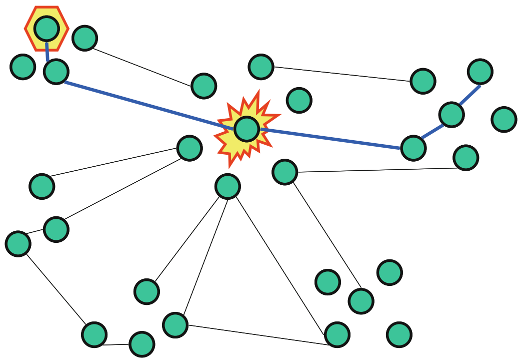
So, which depiction is correct? The answer depends on the purpose of your analysis. What questions do you wish to answer? Different questions lead to sometimes different depictions of the situation – or perhaps more correctly we should say different depictions of the system. Imagine, for example, that you notice a "problem" is identified for one of these items:



It is natural for most people to immediately group the more obvious elements into their presupposed system and then start to do analysis of the situation. The tendency is to identify the cause of a problem to be in nearby things. A system boundary is drawn around the collection of things that appear to be in more or less direct contact with the problem item. But if they happen to choose the wrong system, then the results of their analysis might lead to a wrong, or perhaps less effective, solution to the perceived problem.

I once worked on a large acoustic surveillance system where during a field trial the electronics assembly equipment experienced failures at the interfaces [8]. These interfaces belonged to a particular item and the item was redesigned to help avoid such failures in the future. During subsequent testing, it was discovered that the root cause of the initial interface failures was due to the cable handling equipment design, not the electronics assembly design. We had drawn the boundaries around what we thought to be the system and proceeded to "fix" the problem by changing the design of that system. We would have been better off by defining several different system configurations and then examining each of these to determine various alternative fixes to the problem.

Going back to the example shown before, by taking the mission thread approach (which is only one among the many systems techniques available to the practitioner), this might have lead to perhaps a very different solution. This approach might lead you to a different conclusion about the root cause (the real problem) of the "pain" (the perceived problem):



48.4 System Constituents

Now I would like to examine the nature of a system in terms of its constituents. A "constituent" is an abstract part of something.[2] A constituent is also "An artifact that is one of the individual parts of which a composite entity is made up; especially a part that can be separated from or attached to a system." [9] An artifact is "A man-made object taken as a whole." [ibid]. A system then is made of *abstract* and *man-made* elements. Furthermore, these parts can also be concrete and not made by man (e.g., water used as a coolant in the car's cooling system).

48.4.1 Products Dimension

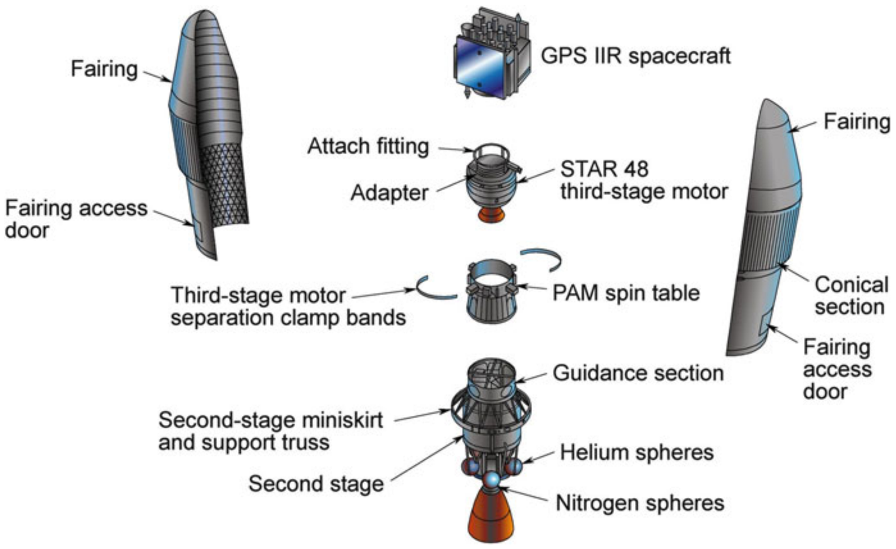
The first type of system component to look at is the thing we call a *Product*. These are things such as hardware and software that we *produce* based on engineering designs. A product is "An artifact that has been created by someone or some process." [9] This collection of products is what most people normally think of as the "system."[3] I would like to extend this notion of products to nonman-made products like water and petroleum. These are *natural* products produced by natural processes. We sometimes will use man-made efforts to convert these natural products to "manufactured" products like bottled water and gasoline (or petrol).

Products can be hardware, software, data, facilities, materials, services, techniques, personnel, and so on. The types of products to be considered depend on the domain being examined. Products will have properties that might be relevant to that product's behavior and suitability with respect to the other products or the overall system. The properties can be logical, physical, relational, conceptual, managerial, and so on. The relevant properties to consider are a function of the purpose of the analysis.

It is very common to depict a system only in terms of its hardware and software components, as illustrated below:

[2]For example, two constituents of a musical composition are melody and harmony. Melody and harmony obviously interact with each in often complex and interesting ways. Notice that the constituents of a system are not necessarily physical entities. In fact, often the most interesting constituents of a system are nonphysical—things like procedures, policies, rules, functions, states, and modes.

[3]When engineers are only considering the products, then this I would call "product engineering" or "product development" as opposed to systems engineering. There is nothing wrong with this for relatively simple products or situations. It is when you have a complex situation or difficult technological issues where you really need true systems thinking.



But this is, by no means, a full depiction of all the various products that make up this system.

So, where do humans come into this schema? Can humans be a product in the system? Notice above where I indicated that one product type could be "personnel," which can be defined as "The body of persons employed by or active in an organization, business, or service." [10] We can consider, of course, that humans are *natural* products conceived in a natural, biological manner, but we can also consider humans to be *manufactured* products of our institutes of education and training.

So why is it inappropriate to think of a collection of products as a system? The so-called system breakdown structure or parts list for the "system" usually only lists the products that belong to the system. It is not wrong, merely incomplete. This type of thinking does not get you very far when you have to understand the systemic nature of a problem, especially when you need to consider how something happening in one part of the system might have an impact (good or bad) on another part of the system. What is missing from this simplistic approach to systems thinking?

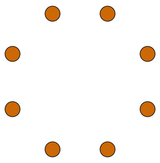
48.4.2 Interactions Dimension

One of the first things to consider beyond the product dimension is the set of *Interactions*. Internal interactions occur between and among the products inside the system. External interactions occur between the products and the outside world (i.e., those things beyond the system boundary). Interactions are often some sort of transfer between objects. The interactions we identify can take on various forms – data flows, material flows, forces, energy, feelings, learning, and so on.

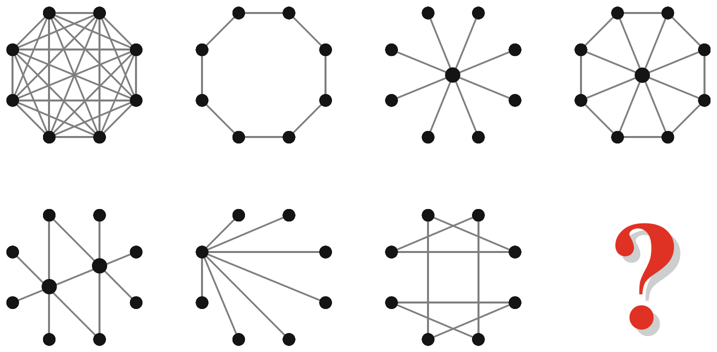
Feelings? How in the world would feelings be relevant when doing systems analysis? Learning? What does learning have to do with systems? Consider what exactly is being transferred between a teacher and student. Is this merely data flow? If so, then this will only result in memorized sets of words and numbers. Is there some sort of energy flow that constitutes the transfer of knowledge? What is being transferred between mother and child, the thing we call love?

In systems thinking, we sometimes must go beyond our common tendencies to think only of the technical flows. But in engineered systems, there may indeed be nontechnical flows to consider, especially when some of the system elements are nontechnical like people.[4]

To illustrate this dimension, let us consider these eight elements:



How many possible ways can they interact? What is the best way for them to interact? What are the advantages and disadvantages of each way?



The answer to these questions is often a matter of *context* . . . .

[4]People, as I mentioned before, are products, too. They are the products of our family situation, our education experience, our social life. People interact with each other and with impersonal objects like hardware and software.

48.4.3 Context Dimension

The third thing to consider when thinking about systems is the *Context*. Context is "The set of facts or circumstances that surround a situation or event." [9] Context consists of environments, scenarios, and situations. These environmental circumstances could be in the form of physical environment (e.g., cold and wet), social environment (e.g., congenial and warm), political environment (e.g., autocratic or democratic), and so on.

The context could describe the particular scenarios of interest to be examined. Will this system need to operate only during good weather or must we also consider bad weather?

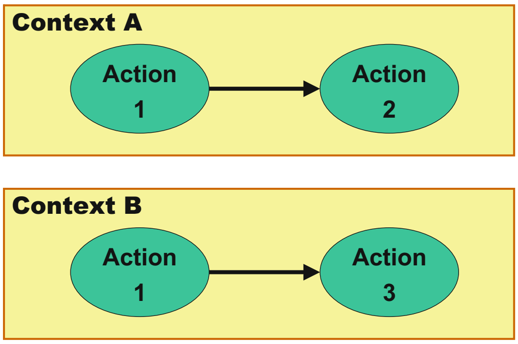
Scenarios deal with postulated sequences of possible events such as wartime where the adversary has already breached the first line of defense, a new market with little or no competition where our new product launch is about to start, economic depression followed by government breakdown along with social unrest, and so on.

Factual situations could be in the form of technology limitations, funding constraints, legislative mandates, human restrictions, physical laws, and so on. Sometimes the situation is not necessarily "factual" as there may not be clearly established "facts" but merely "beliefs" – what people believe is true but may or may not actually be true. When examining a system for proper functionality and behavior that includes people, it is often more important to understand what the people believe than what is in fact true.

Hardware and software are often more directly connected to the factual situation as they usually get their data from objective sensors, verified processors, and validated databases. People, on the other hand, have "subjective" sensors and often nontransparent processing algorithms (i.e., thinking). Their thinking might be crisp and logical, or it might be soft and fuzzy. Nonetheless, the behavior of the people in this setting could have significant impact on the proper behavior of the system of interest.

48.4.4 Actions Dimension

The fourth thing to consider when thinking of systems is the identification of desirable (and possibly undesirable) *Actions*. Action of the products is what causes the *interactions* within different contexts to occur. Action is how the products respond to actions by other products. The action and reaction are what constitutes the interaction we considered above. Action is what each product itself does, and interaction is what the products do to each other. Sometimes a certain action by a product will cause a different interaction with another product depending on the particular *context*. In other words, the appropriate action in each case is context-dependent.



The action dimension of systems analysis has a long history in the discipline of systems engineering. Traditionally, it is called functional analysis, but lately there is a relatively new approach being used in systems engineering called object-oriented analysis. Even more recent is an approach called service-oriented analysis. There is not a single approach that is best for all situations. The approach chosen may be dependent on the types of products that dominate our system. For example, software-dominant systems analysis might need object-oriented techniques, while people-dominant systems analysis might use the service-oriented approach.

48.4.5 Relationships Dimension

Relationship is a connection, association, or involvement between two or more things. An electronic circuit can be mounted *inside* a chassis. A startup routine must occur *before* normal operations can be performed. An ultraviolet transmission occurs *above* the normal range of visible light. A *Relationship* can be spatial, temporal, or spectral. It can be social, political, or organizational. Relationships often dictate what is allowed to happen or what must happen.

If a country has a treaty with another country, then certain restrictions apply that inhibit certain actions. If Susan's mother is on the organizing committee, then this might affect whether Susan gets the appointment (and likely will affect how Susan behaves after receiving the appointment). If an antiaircraft weapon is placed under the camouflage net, then this impacts its observability from above (and likely will increase its survivability under battlefield conditions). As you can see, relationships are not the same as "interactions," but they can have a definite impact on what, when, and how interactions occur.

Relationships are the most important aspect of systems thinking. Without relationships, we cannot even *imagine* the system properly. However, we tend to limit our thinking only to the *interfaces* between products where we have flow of data, material, or energy. We must expand our repertoire of relationships beyond just interfaces. By doing so, we can expand our ability to perform better systems thinking.

48.4.6 Destiny Dimension

Destiny is the "predetermined, usually inevitable or irresistible, *course of events*" [11] that will befall the system. However, *Destiny* is not only what happens to the system over time. It is also the resultant *outcome* of having the system in place. These outcomes ultimately determine the *value* of the system. So, destiny is the ultimate reason for bringing the system into existence.

Some might argue that you should only worry about the *purpose* of the system as the ultimate consequences of using that system are beyond our control. However, we need to consider the various possible ways that our system can be used, misused, and abused as these could have negative consequences that negate all the positive benefits we had originally in mind when setting out to design the system for a defined "purpose." Its destiny is about more than just knowing the purpose (or function) of the system. We must also consider the wide and sometimes far-reaching consequences of the system during (and even after) its lifetime of operation.

The aim of systems engineering should be to influence the destiny of a system and perhaps even be the prime motivator to make the perceived course of events come about. Sometimes destiny is referred to as the "mission" of the system. We often define this in terms of a "concept of operations" for the system. We translate this into a set of requirements that specify the full range of behavior and performance of the system to ensure that it will fulfill its "preconceived" destiny.

Systems engineering never has a completely accurate crystal ball, so there is always some element of surprise when the destiny of the system is finally realized. But, nonetheless, we must keep in mind what Plato had to say about this: "Begin with the end in mind."

48.5 Using the PICARD Theory for Enhanced Holistic Thinking

It is by examining these six types of things that true systems thinking can occur. The PICARD theory of systems (products, interaction, context, action, relationship, and destiny) encompasses the connections between the PICARD elements of the system are what give the system its "emergent" behavior. Each product within a system will have its own behavior, but this behavior is often of little use by itself until it creates a system-level behavior that provides a useful feature or function to the system user or operator.

System = Products
+ Interactions + Context
+ Actions + Relationships + Destiny

48.6 Conclusions

This, in the end, is what systems engineering is all about: translating user needs into a collection of engineering products to be built using technology elements (as well as sometimes including nontechnical elements such as people, practices, policies, organizational constructs, etc.).

Systems engineering is, in other words, the process by which we turn the imaginary (concept) into the real (products, services, etc.). The degree of "realness" depends on how far we have got in managing the risk as we progress and what intermediate "artifacts" we produce to get us there (e.g., requirements documents, architecture diagrams, test specifications, prototypes). It is all about transition from the imaginary to the real [12]..

But the trick is not going directly from stated needs to product attributes. We must consider through the Systems Engineering (SE) process how interactions, context, and actions help the products meet those user needs. Furthermore, we must properly consider how relationships impact the overall behavior and how the destiny of the system can eventually be achieved. Using the PICARD theory elements can be helpful for anyone on their "journey through the systems landscape." [13].

We can facilitate the systems engineering process by having better systems thinking. The PICARD theory of systems holds some promise in helping us think more precisely about the systems we intend to engineer. It will hopefully help us avoid the mistakes that can result from incompletely or incorrectly defining the appropriate system of interest.

We tend to think our systems are *real*, but in reality, they are what we *think*.

Biography

James Martin is an enterprise architect and systems engineer affiliated with The Aerospace Corporation developing solutions for information systems and space systems. He was a key author on the BKCASE project in development of the SE Body of Knowledge (SEBOK). His main SEBOK contribution was the articles on Enterprise Systems Engineering. Dr. Martin led the working group responsible for developing ANSI/EIA 632, a U.S. national standard that defines the processes for engineering a system. He previously worked for Raytheon Systems Company as a lead systems engineer and architect on airborne and satellite communications networks. He has also worked at AT&T Bell Labs on wireless telecommunications products and underwater fiber optic transmission products. His book, *Systems Engineering Guidebook*, has been widely used to help in implementing the systems practice. Dr. Martin is an INCOSE fellow and was leader of the Standards Technical Committee. He received from INCOSE the Founders Award for his long and distinguished achievements in the field.

References

1. Sowa JF (2000) Knowledge representation: logical, philosophical, and computational foundations. Brooks/Cole Thomas Learning, Pacific Grove
2. Weinberg GM (2001) An introduction to general systems thinking, 1st edn. Dorset House Publishing Company, New York
3. Hitchins DK (2004) Advanced systems thinking, engineering, and management. Artech House, Boston
4. Van Daalen, C. Els, Wil A. H. Thissen, and Alexander Verbraeck, "Methods for the modeling and analysis of alternatives,Sage, Andrew P. Rouse, William B, Handbook of systems engineering and management, New York John Wiley & Sons, 1999.
5. Wasson CS (2006) System analysis, design, and development: concepts, principles, and practices. John Wiley & Sons, Hoboken
6. Blanchard BS, Fabrycky WJ (2005) Systems engineering and analysis, 4th edn. Prentice Hall, Upper Saddle River
7. Hall AD (1989) Metasystems methodology: a new synthesis and unification. Pergamon Press, Oxford/New York
8. Martin JN (1993) Reverse engineering a system using life cycle modeling, Proceedings of the NCOSE International Symposium, 1993
9. www.wordwebonline.com
10. (2006) American heritage dictionary. 4th edn. Houghton Mifflin, Boston
11. www.dictionary.com
12. Paraphrased from a message by Trevor Rickard at QinetQ to me in regard to a prior version of this paper.
13. Lawson H(B)W (2010) A journey through the systems landscape. College Publications, London

Chapter 49
Agency and Causal Factors in Social System Behavior: Advancing Human Systems Engineering with General System Theory

Susan Farr Gabriele

Abstract In spite of significant advances in technology in today's world, our large social systems are marked by increasing social decline. A human systems paradigm can inform and be informed by analysis and clarification of the hard facts of our soft social systems. This chapter aims to identify the flaws in practice and theory underlying our current social systems, and then correct them using a wider knowledge base gathered from relevant disciplines. Updated theory is that agency of organization behavior is not in the leader, nor the worker, but in both. Each system member learns and performs according to his/her own willingness and ability, resulting in almost infinite variability. Thus, a new provide/pickup paradigm is proposed. The leader's role is to provide input, resources, and tasks; the learner/worker role is to pickup input, each at his/her own rate. In large social systems, important input is beyond the pickup range of individuals. User-designed, ideal-based automated social control systems are proposed to allow organizations and system members to flourish.

Keywords Systems and SoS integration • Cognitive systems engineering • Systems thinking and complexity management • Infusion of systems science in systems engineering • Systems engineering and decision science

49.1 Background: Declining Outcomes in Large Social Systems

Science offers useful laws for how *things* behave, or the *hard* sciences, such as chemistry, physics, math, and engineering. We know how to make water of two parts hydrogen and one part oxygen. We know about the laws of gravity. We know that 19 + 1 = 20. We know how to design complex mechanical control systems, such as office thermostat systems and guided missiles. On the other hand, science

S.F. Gabriele (✉)
Gabriele Educational Materials and Systems, Valencia, CA, USA
e-mail: sgabriele@gemslearning.net

A.M. Madni et al. (eds.), *Disciplinary Convergence in Systems Engineering Research*, https://doi.org/10.1007/978-3-319-62217-0_49

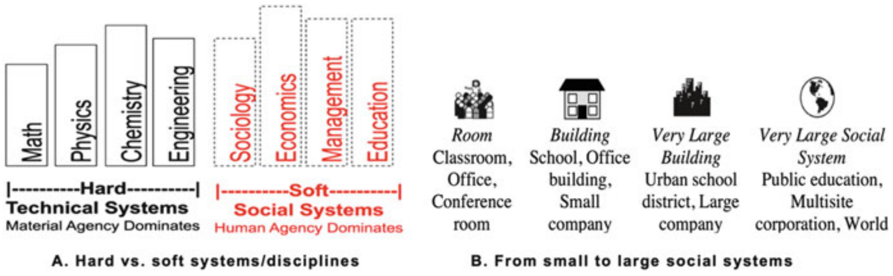

Math
Physics
Chemistry
Engineering
Sociology
Economics
Management
Education
|----------Hard----------|
Technical Systems
Material Agency Dominates
|----------Soft----------|
Social Systems
Human Agency Dominates
Room
Classroom, Office, Conference room
Building
School, Office building, Small company
Very Large Building
Urban school district, Large company
Very Large Social System
Public education, Multisite corporation, World
A. Hard vs. soft systems/disciplines
B. From small to large social systems

Fig. 49.1 Views of systems and social systems. (**a**) Hard vs. soft systems/disciplines. (**b**) From small to large social systems

offers few and conflicting models for how *people* behave. Thus, there are the *soft* sciences, such as psychology, management, education, sociology, and economics. And there are the *soft* social systems, such as schools and workplaces (Fig. 49.1).

Over the last half century, considerable progress has been made in technology and equity in our large social systems. However, important dimensions have not kept up and our large social systems are in increasing decline. This has resulted in two outcomes captured in two images: the *Tower of Babel effect* and the *19 + 1 = 18 effect*, explained in next section. Examples come from two large institutions: public education and workplaces.

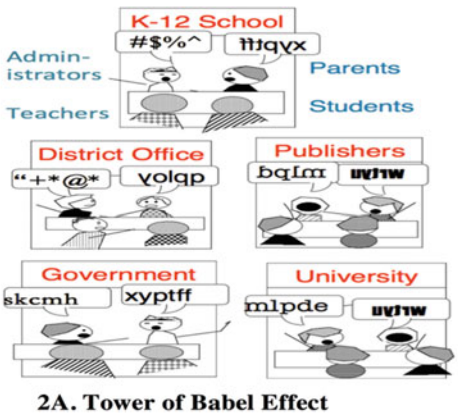
49.1.1 Schools

Public education is currently troubled by these two outcomes. Lack of collaboration time, plus all the differing viewpoints, especially at the various system levels (i.e., classroom, school, school district, state/federal departments of education) leave school decision makers unable to understand each other, resulting in the *Tower of Babel effect* (Fig. 49.2a).

Ever-increasing demands (Fig. 49.2b, shaded circle) leave teachers less able to address their students' needs, so school quality goes down, illustrated in the bottom clockwise cycle. The top counterclockwise cycle shows teachers leaving the classroom, perhaps leaving public education altogether, or to become administrators. In both cycles, desperate new policies are mandated too quickly for schools to keep up with. The result is *the 19 + 1 = 18 effect: 19* (school quality) + *1* (new demand) = 18 (reduced school quality). Over 3 years, the process looks like 19 + 1 = 18, 17, 16 [8].

Other educational scholars have similar findings. Sarason authored a book entitled *The Predictable Failure of Educational Reform* [15]. Silverman wrote that

> ... The reason the reform movement [in the 70's] failed was 'the fact that it's prime movers were distinguished university scholars'; . . . what was assumed to be its greatest strength turned out to be its greatest weakness . . . well-intentioned intelligent university authorities and 'experts' on education can be dead wrong. The reforms failed because of faulty and overly abstract theories not related or relatable to practice, limited or no contact with an understanding of the school (Silverman in Fullan [7], p. 22)



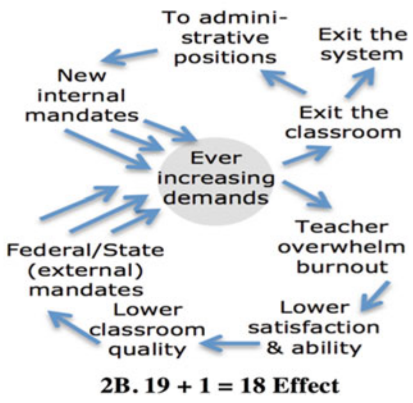


Fig. 49.2 Two unintended outcomes. (**a**) Tower of Babel Effect. (**b**) 19 + 1 = 18 Effect

49.1.2 Workplaces

Many large workplaces have the same challenges. Bolman and Deal, researching organizations, reported the following incident.

> "We were once talking to a group of managers in a company with an extensive MBO [Management by Objective] program, and we asked them how MBO was working. The first answer was:
>
> "We don't have MBO. We have MBT."
>
> "What is MBT?" we asked.
>
> "Management by terror." ([2], p. 80)

The Space Shuttle Challenger disaster is another example. According to the Rogers Commission, NASA's organizational culture and decision-making processes were key contributing factors to the accident, with the agency violating its own safety rules. NASA managers had failed to correct a potential design flaw, and engineers had failed to adequately report their concerns (Wikipedia).

There is evidence of increasing decline in large social systems worldwide. In fact, human systems engineering is a field that came out of the crisis of the Swiss banking system and the findings that "human risks" are a major problem in organizations (Wikipedia, 2015). If these undesired outcomes, the 19 + 1 = 18 and Tower of Babel effects, are indeed reflective of large social systems of many types worldwide, then there is hope! It shows that there is predictability and that there are scientific laws at work in social systems. It is just that the underlying laws have not been fully specified.

49.1.3 Aim of this Chapter

This chapter gleans out the hard facts, root causes, or agency of learning and behavior in organizations. Causes clarified, updated theory, practice, and solutions are proposed for the design, engineering, and management of flourishing, evolving social systems to be illustrated in a new 3-year metaphor: 19 + 1 = 20, 21, 22.

49.2 Methodology

The process used to investigate these issues is narrative path analysis. Beginning with large social systems as the unit of focus (top left in Fig. 49.3), the path proceeds down to identify flawed practices, then underlying theory and assumptions landing at the individual human system member as unit of focus. Conflicting theories are unified and updated. The narrative path starts a return up to the very large social system, offering updated practice. The discussion is informed by key concepts, literature, and evidence from instruction, management, and especially general systems theory and design. Other fields that enrich this discussion include control systems engineering, psychology, adult learning theory, plus examples from large urban schools and workplaces. Clarifying images are offered to allow discussion of details or examples along with the more grand-level principles, with the goals of making sense to a wide diverse audience. Figure 49.3 presents the path in a nutshell in a U. The following sections develop the path.

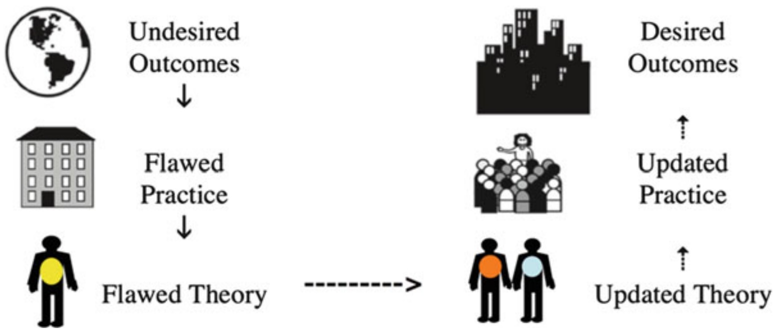


Fig. 49.3 Methodology illustrated in a nutshell

49.3 Flawed Practice

Old paradigm, traditional, or hard science thinking, illustrated as $19 + 1 = 20$, does not apply to social systems. A new paradigm is needed. However, efforts at detailing a new paradigm are muddled, resulting in two conflicting paradigms and practices – one is often known as the *top-down directive* old paradigm and the other known as the *bottom-up participatory* new paradigm. *Old paradigm* leaders might see the undesired outcomes. They try to improve their organizations by increasing their top-down efforts. *New paradigm* leaders realize that their students or staff all have different learning rates. They might overcorrect, giving too much flexibility to employees or learners, resulting in the laissez-faire approach. The not-fully-specified new paradigm leader is unsure of his/her role.

49.4 Flawed Theory

Hidden under the flawed practices are conflicting assumptions and theory. Old paradigm leadership assumes sole agency or cause of organization behavior is in the leader. New paradigm leadership assumes sole agency or cause of organization behavior is in the learner or employee. This is the either/or dilemma underlying much current conflicting, confused practice.

49.5 Updated Theory and Practice

The first step in the path to the more fully specified new paradigm is the shift in agency – from teacher to learner, from CEO to employee. This shift is as dramatic and far-reaching as the earth/sun rotation paradigm shift in astronomy. Whether behavioral laws and causes relate to gravity or human agency, both paradigm shifts here are proposed as hard science – a result of extensive empirical observation, rather than speculation. A shift at such a grand level requires reconceptualization and recalculation at all levels of system.

The shift in instruction/management theory is only a partial answer, resulting in the two conflicting camps: those who propose that the leader is sole agent and must control the supervised versus those who argue that the supervised are agents of their own learning/performance and need total flexibility. In this chapter, satisfying resolution is proposed in an elaboration of Kenneth Boulding's general systems theory [4, 8, 9].

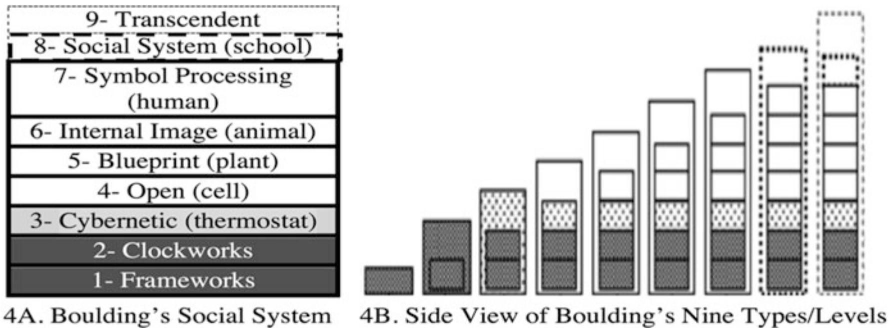

9- Transcendent
8- Social System (school)
7- Symbol Processing (human)
6- Internal Image (animal)
5- Blueprint (plant)
4- Open (cell)
3- Cybernetic (thermostat)
2- Clockworks
1- Frameworks
4A. Boulding's Social System
4B. Side View of Boulding's Nine Types/Levels

Fig. 49.4 Boulding's nine level typology and social system. (**a**) Boulding's Social System. (**b**) Side View of Boulding's Nine Types/Levels

49.5.1 Boulding's General System Theory

Kenneth Boulding, a cofounder of general system theory, looked to nature to uncover the hard facts of soft social systems. He organized the systems of the world into his typology of system complexity (Fig. 49.4a). Each type is composed of all the levels below it (Fig. 49.4b) and is named by the new property that it adds (Fig. 49.4a).

Boulding's nine-level social system unifies the conflicting camps. In other words, the top-down bureaucratic models assume all parts of a social system are designable. Laissez-faire models assume no parts are designable. Boulding's typology shows how both paradigms have merit and which parts of a social system are designable and which are not. Frameworks, clockworks, and control systems or "thermostats" (Levels 1–3 in Fig. 49.4) are predictable and designable to *exteriorly prescribed criteria* (e.g., goals determined by a teacher, engineer, or CEO). Open, blueprint, image-aware, and symbol-processing parts (Levels 4–7) are not designable. These undesignable systems, organisms, act according to *interiorly prescribed criteria* – needs (Level 4: e.g., living cell), abilities (Level 5: e.g., plant), perceptions (Level 6: e.g., animal), and choices (Level 7: human) – of increasing variability. Level 7 system boundaries are mandatory; Level 8 is optional (illustrated with dashed lines). Social and transcendent levels (Levels 8–9) are even more variable. Level 7 systems (humans) can ignore the leader's input and even take opposite action. Thus, Level 7 (individual) goals preempt Level 8 (organization) goals. Individual humans can move from one Level 8 system to another – changing their schools or workplaces. They cannot change their Level 7 system – their physical body.

49.5.2 The Individual Human Being as Agent

Boulding's typology reveals that each system member is agent of his/her own learning and behavior. This is true of all people in the system, both the supervised and their leaders or supervisors. Boulding's social system informs TPO theory (i.e., Things, People, Outcomes), which clarifies the following principles. In a social system such as a school or organization, the leader's tasks, policies, and resources or *THINGS* (Levels 1–3) will be used by *PEOPLE* in the system to meet their own self-determined needs and goals (Levels 4–7), according to their own individual differences, whether inherent or learned (Level 5), their own immediate perceptions from among conflicting stimuli (Level 6), and their short- or long-term choices (Level 7). It is a natural hard scientific fact (physics, not ethics) that Level 7 systems, *PEOPLE,* must adequately meet their basic individual needs (survival, safety, and belonging) before the needs of the organization (Levels 8–9), which determines *OUTCOMES*.

Implications for social system design are as follows: In a social system, such as a school or other organization, *THINGS* (Levels 1–3: resources, equipment, materials, schedules, policies) must be designed and arranged so that *PEOPLE*, each at his/her own pace, can easily meet both their self-determined individual goals (Levels 4–7) and their organization's goals for best *OUTCOMES* (Levels 8–9).

49.5.3 Within the Individual: Pickup, Throughput, Output, and Links to Boulding

A final downshift in focus lands on the individual. Figure 49.5 illustrates the structures and processes of pickup and output and those in between (throughput). The unit of focus is the individual. Figure 49.5a illustrates three main pickup points (in red): the eyes, ears, and hands.

Figure 49.5b downshifts from outside the individual to inside the individual. Pickup occurs when there is an adequate match of the input to the individual's CAP domains – cognitive (dark gray), affective (yellow), and psychomotor (light gray). Depending on each individual, it may be followed by learning, mastery, creativity, and action/performance. If there is not an adequate match or serious mismatch, the individual may not notice, ignore, misinterpret, or display fight, flight, or submit responses.

Figure 49.5c upshifts from inside the individual to outside the individual again. Pickup is followed by individually variable throughput, and then results in even more variable outputs. Figure 49.5c illustrates three main output points (in red): the mouth, hands, and feet.

Links to Boulding in Fig. 49.5 are as follows: Level 1 frameworks in the pickup through output processes are eyes, ears, hands, mouth, and feet; and also, inside the individual, the CAP domains. Pickup, when automatic, is mainly a Level

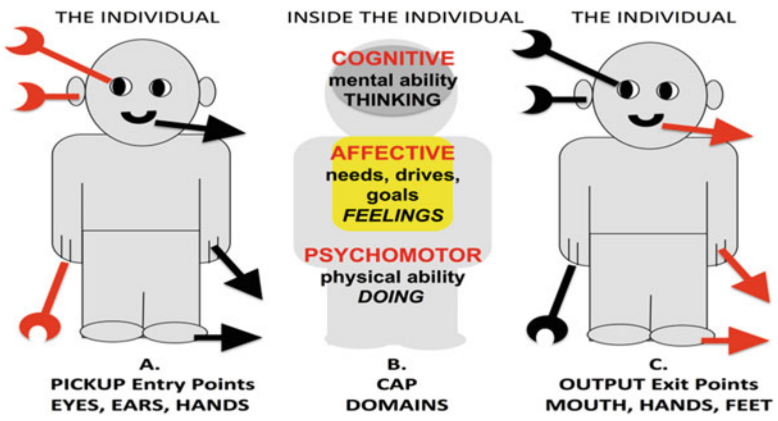


Fig. 49.5 From pickup to output at the level of the individual. (**a**) Pickup entry points eyes, ears, hands. (**b**) Cap domains. (**c**) Output exit points mouth, hands, feet

2 clockwork process as are circulation, respiration, and digestion. Level 3 is a control system, an ON/OFF switch. When there is a CAP match, the process is ON and pickup occurs. When there is a CAP block, the process turns OFF and pickup does not occur or is skewed. Levels 4–7 add nonclockwork processes determined by interiorly prescribed criteria. In other words, at Level 7, pickup is determined by each individual's image, his/her willingness (affective), and ability (cognitive and psychomotor). Throughputs and outputs are nonclockwork.

In other words, pickup occurs when there is adequate match of the input (what the leader provides) to the learner's CAP domains. To be clear, pickup is just a first step in the task of the system member – learner, worker, or engineer. The individual system member, in the process of learning and performance, will pick up, learn, and master the input. He/she then may act, perform, or create a corresponding product. Main entry points for pickup are the eyes, ears, and hands. Main exit points for outputs are the mouth, hands, and feet/body. The focus here is pickup, however, because pickup is where the breakdown occurs, elaborated in the next sections. It is important to reiterate that pickup will not occur if there is a block in any of the domains. For example, a student or employee may not understand the task (cognitive), or he or she may not see value in the task (affective), or he or she may feel overloaded with too many other tasks to do and does not notice or retain the new task (psychomotor) .

49.5.4 Updated Practice: Room Level

Upshifting to the room level, informed, experienced leaders (teachers, facilitators, and managers) aim to create an environment with many opportunities for pickup. Leaders may usefully compare the systems that they supervise to a complex

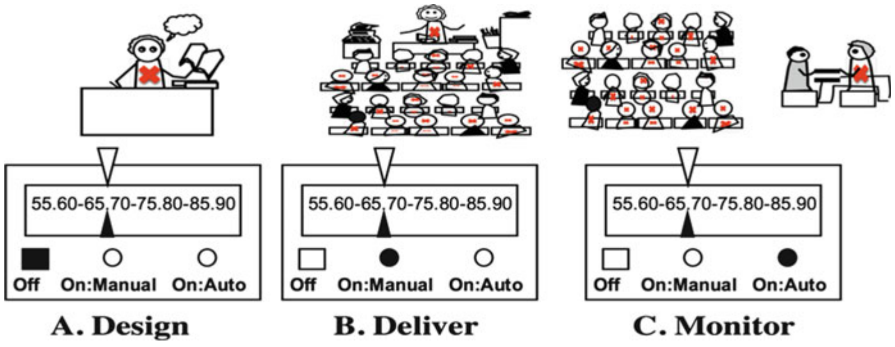


Fig. 49.6 Three mode thermostat leadership. (**a**) Design. (**b**) Deliver. (**c**) Monitor

thermostat system (Fig. 49.6) with three modes – design (c.f., OFF), deliver (c.f., ON: Manual), and then monitor (c.f., ON: Auto). Instead of goals of optimal range of temperature, heat (65–75 degrees), their goals are optimal CAP, or input that is in a range with system members' cognitive, affective, and physical/psychomotor domains. When work or class is not in session, the leader or leadership team designs the input and resources. Metaphorically, windows and doors can be wide open, as the "heater" is turned off so heat (resources) will not be wasted out the window (Fig. 49.6a). At the beginning of a project or school semester, when work or class is in session, the leader delivers the input, introducing the new input and carefully managing the delivery, keeping the room at a range that matches the CAP of the learners (c.f., keeping the temperature range of 65–75 degrees). Metaphorically, the heat is turned on and being distributed throughout the room. Windows and doors are closed, so resources are not lost out the window (Fig. 49.6b). Nor are disruptions coming through open windows. When learners have picked up and acquired the new input to a sufficient degree, and everyone is on task, the leader shifts to ON: Auto. Learners and workers continue with their tasks independently. Leaders are then freed up to do their own work (Fig. 49.6c).

In ON: Auto, leaders also monitor the room to adjust the providing and to notice if someone is off-task, where pickup has not occurred, to determine or help the system member identify the block preventing pickup. A block might be cognitive: For example, the learner or worker does not understand the task. It might be affective: For example, he/she does not see the importance of the new task and has set it aside to continue other work. A block might be physical/psychomotor: For example, he/she needs glasses and cannot read the small font of the document. It might be a mixture: For example, the worker did not eat breakfast, cannot concentrate, and also thinks the project is unimportant, not useful, or even flawed [8].

This ON: Auto mode, where the leader puts his energy in his/her own work while remaining aware of the work environment, has an interesting parallel in a best practice in business called *management by exception*, which is:

A style of management that involves giving the people who work for you the authority to control their work or particular jobs, projects, etc., unless there is an exception (= an unusual situation) that causes a problem [6].

49.5.5 Updated Practice for Large Social Systems

The final step on the path is the upshift in the unit of analysis from the room (classroom or work team) or small building, to the very large, multisite corporation, institution (e.g., public education), or other social system. At this point, the term *span of control* serves to introduce the important new issues that arise. Span of control is a term used commonly in business management, referring to the number of subordinates a supervisor has. It is most closely related to the old paradigm assumption of teaching and management: leader as sole agent. The term *span of control* can be usefully reconceptualized to *span of pickup* or *span of CAP pickup* to fit the more fully specified paradigm – learners as agents, everyone a learner, and the infinitely variable learning and behavior of individual members of social systems. This human systems paradigm, that understands agency in the individual, that the first step in learning is pickup, undergirds this new term. At the room or small building level, CAP identifies the *nature* of pickup or a block in pickup. The nature of pickup is the fact that the individual will pick up (learn and master) according to the match of the input with his/her unique CAP domains. *CAP span* refers to another dimension of pickup, its *range*. The range of pickup is a key new issue in large social systems, where input may not be in the range of the system member's (1) awareness and understanding (cognitive span), (2) concern and care (affective span), and (3) physical control (psychomotor span).

Hardin's "Tragedy of the Commons." Range of pickup, or CAP span, is a significant issue in large social systems. Garrett Hardin addresses this very issue in his seminal paper. Hardin's *Tragedy of the Commons* [10], using the example of cattle herders and grazing lands, explains how individual herders will overuse common pool resources (CPRs) because they easily see the advantages for their own personal gains, but are too distant from the big picture, too distant from the toll it takes on all the others in the system. With regard to the terms introduced here, Hardin found the CAP span insurmountable, or in terms introduced in this chapter, that pickup was outside the individual CAP range. Hardin further argued that there was no technical solution to such grand problems.

Ostrum's "Revisiting of the Commons." On the other hand, Ostrum and colleagues found evidence that institutions can successfully govern CPRs, especially when "individuals face a public good or CPR problem and are able to communicate, sanction one another, or make new rules" (1998, p. 279). In other words, Ostrum found the CAP span surmountable, that individual system member CAP pickup was possible, given certain conditions – such as common goals, mutual respect, and ability to communicate. Ostrum's findings are clarified for large social systems by insights from James Martin, the leader of our INCOSE system science working group. Martin brought attention to the multiple levels of organization in a large social system and the fact that, at each level, specific expertise is different and resides within members of the specific level. He explained that a specific solution to a problem should be designed by members of the specific system level or type and then approved by the level immediately above it [11].

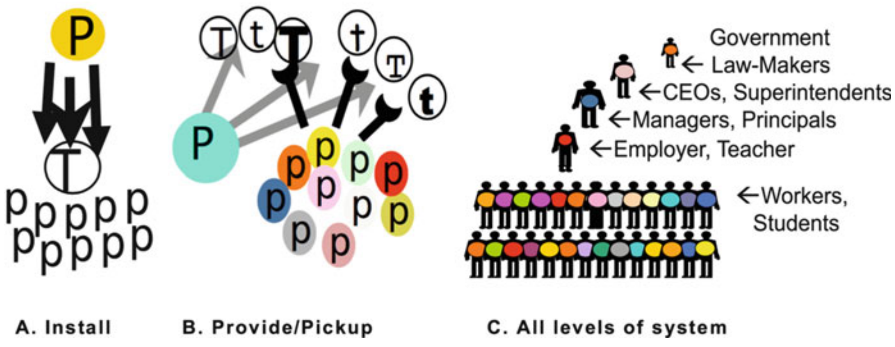


FIg. 49.7 Agency and infinite variability of learning and behavior. (**a**) Install. (**b**) Provide/pickup. (**c**) All levels of system

49.6 Conclusions

Social systems learn and behave not by system leader *installation* (illustrated by arrows, left in Fig. 49.7a), but by system member *pickup* (illustrated by graspers, right in Fig. 49.7b*)*. Pickup occurs if the leader's provided input, resources, or display (T) matches system member's CAP domains (illustrated in Fig. 49.5). The great variability in every learner is indicated by colors (Fig. 49.7). Thus, there is value in providing and displaying tasks, input, and resources in a variety of ways (illustrated by differently shaped Ts). Two paradigms are illustrated in Fig. 49.7b. Left, the old paradigm: Agency in the leader (P), rather than the employees or learners (pp). Center, the updated *provide*/*pickup* paradigm.

Pickup, which has been suggested as infinitely variable in individuals, is even more variable due to system levels, as each level has different functions and all system members are learners (Fig. 49.7c). Figure 49.7c illustrates the infinite variability – in learners, both system members and system leaders, as well as in system levels and types.

In large social systems, much important input is beyond system members' pickup span. For example, it is easier for CEOs in the ivory tower to care more about their children's college tuition than their employees' salaries. And, it is easier for front-line employees to care more about their weekly paycheck than the big picture goals of the organization.

49.7 Follow-on Work

Follow-on work suggested is informed by Boulding's Level 3 control system as the key to social system agility and heath. Principles of mechanical control systems, plus principles of provide/pickup, TPO theory, and CAP can inform social control systems. There is merit to an investigation and elaboration of the *Thermostat Leadership* metaphor and *User-Designed Ideal-Based Automated Social Control Systems.*

49.7.1 Thermostat Leadership

First, system leaders, with the members of their systems, are to specify details of the provide/pickup thermostat metaphor for their particular social system and system level. For example, they are to identify and carefully design, deliver, and monitor their frameworks, clockworks, and control systems for optimal social system function, to increase opportunities for optimal system member pickup, to increase likeliness that, over 3 years, the result will be $19 + 1 = 20, 21, 22$. In a mechanical control system such as a thermostat system, size of the building, size of the engine, number of vents, placement of vents, and so forth, and their relationship to each other (their ratios) are key to its effective functioning. Ratios are key for effective control systems and for effective social control systems.

In some domains of some large social systems, optimal ratios are policy. For example, the California Education Code [5] states

> "41,400. It is the intent and purpose of the Legislature to improve public education in California by maximizing the allocation of existing resources, to discourage the growth of bureaucracy in the public schools, and to emphasize the importance and significance of the classroom teacher."
>
> "41,402. The maximum ratios of administrative employees to each 100 teachers in the various types of school districts shall be as follows:
>
> (a) In elementary school districts—9.
> (b) In unified school districts—8.
> (c) In high school districts—7."

In other cases, optimal ratios are not policy. The ratio of a CEO's salary to worker salaries might be an example. Corinne Wilson reported on the Institute for Policy Studies in Washington's 18th annual survey of executive compensation. Her findings were that the "263-to-1 ratio between CEO pay and average worker pay in the U.S. grew to 325-to-1 last year" ([16], p. 1). Wilson further argues that "our communities will thrive when we bring the unemployed and underpaid into the middle class, so they can pay their mortgages and their taxes" ([16], p.1).

CEOs and leadership teams, each at their own system level, would do well to develop holistic, systemic perspectives of their organization, to understand their functioning and choose their optimal ratios for optimal outcomes, to achieve the $19 + 1 = 20, 21, 22$ effect.

49.7.2 User-Designed Ideal-Based Automated Social Control Systems

Second, users are to automate the desired ratios. The 8:100 ratio of administrator to teacher and ideal ratio of CEO salary to employee salaries could be linked to payroll. It has been 50 years since Hardin wrote that there was no technical solution to the tragedy of the commons. Today, we do have the technology. Today, it can be accomplished. However and moreover, it is to be accomplished by the users themselves, at their own level of system, within the policies of the larger system in which it is embedded.

A word of caution: It is important to clarify an empowering rationale for *user-designed automated social control systems*. Linking user-determined optimal ratios (e.g., leader/employee ratios and salaries) to payroll is not to criticize, punish, or weaken current leaders or any system members (e.g., the cow herder). On the contrary, it is to free up system member energy. McPherson illuminates an important principle here, claiming that "neither the few destructive laggards nor the handful of brilliant performers" are the key to organization health. Instead, he urges attention to the "care, feeding, and unshackling of the average man" ([14], p. xxii).

The value of automated policy consequences recalls the findings of Berliner (1986), who found an abundance of "scripted" review routines in his observations of expert teachers' classrooms. He found routines

> ... embedded in the classroom activities ... shared, scripted, virtually automated pieces of action [that] allow students and teachers to devote their attention to other, perhaps more important, matters inherent in the lesson ([1], p. 5).

User-designed ideal-based automated social control systems are to allow leaders and system members at each level of system to design their own optimal "thermostat" systems – including types and flows of resources. Automation is to bring the important big picture policy into system members' pickup range, to free their attention for more important matters.

In a nutshell, the elements or cumulative meaning of *user-designed ideal-based automated social control systems* is constructed using the following examples.

Control systems → When the temperature turns 65, the heater turns on.

- *Social* → When an employee is late, he/she makes up the time (honor system or superviser controlled).
- *Automated* → When an employee is late, he/she makes up the time (information automatically goes to time clock and payroll).
- *Ideal-based* → The aim is not to berate or punish, but to free up everyone's time for more important matters.
- *User-designed* → People at each system level decide together the automated consequences for themselves (in alignment with suprasystem policy).

References

1. Berliner D (1986) In pursuit of the expert pedagogue. Educ Res 15:5–13
2. Bolman L, Deal T (1990) Modern approaches to understanding and managing organizations. Jossey Bass, San Francisco
3. Bott P (1995) Testing and assessment in occupational and technical education. Allyn & Bacon, Boston
4. Boulding K (1956) General systems theory: the skeleton of science. Manag Sci 11(3):197–208
5. California (2015) California education code. Teaching and nonteaching certificated employee ratios. Sections: 41400–41409.3. Retrieved December 7, 2015., from http://www.leginfo.ca.gov/cgi-bin/displaycode?section=edc&group=41001-42000&file=41400-41409.3
6. Management by Exception (2016) Retrieved on June 14, 2016, from http://dictionary.cambridge.org/us/dictionary/english/management-by-exception
7. Fullan M (1991) The new meaning of educational change. Teachers' College Press, New York
8. Gabriele S (2014) New hope for schools: findings of a teacher turned detective. iUniverse
9. Gabriele S (1997) Boulding's typology elaborated: a framework for understanding school and classroom systems. Syst Pract 10(3):271–304
10. Hardin G (1968) The tragedy of the commons. Science 162(3859):1243–1248
11. Martin J (2015) Systems thinking workshop: learning to think about systems in a holistic manner. Redondo Beach, California
12. Ostrum E, Burger J, Field C, Norgaard R, Policansky D (1999) Revisiting the commons: local lessons, global challenges. Science 284:279
13. Patterson K, Covey S (2002) Crucial conversations: tools for talking when stakes are high. McGraw-Hill, New York
14. Peters T, Waterman R (1982) In search of excellence. Harper and Row, New York
15. Sarason S (1993) The predictable failure of educational reform: can we change course before it's too late. Allyn and Bacon, New York
16. Wilson C (2011) Opinion: difference between CEO and worker pay is unconscionable. Voice of San Diego. Retrieved June 8, 2016, from http://www.voiceofsandiego.org/topics/opinion/opinion-difference-between-ceo-and-worker-pay-is-unconscionable

Chapter 50
Classifying Emergent Behavior to Reveal Design Patterns

Jack B. Reid and Donna H. Rhodes

Abstract Methods for breaking down emergent phenomena into categories typically focus on some measure, qualitative or quantitative, of the degree of complexity of the system. While such categories are useful for clarifying what, exactly, is meant by the word "emergence," they are less useful for developing practical means of identifying, mitigating, or encouraging emergent behavior. This chapter discusses several systems of classification of emergence that focus instead on characterizing either the form of the emergent behavior or the causal factors that encourage it. These typologies are used as a basis to propose the development of a set of design patterns that are broadly applicable to designing for emergent behavior in systems of a variety of domains. Case studies are used to illustrate these principles, and further work toward expanding and codifying the principles is discussed.

Keywords Emergence • Emergent phenomena • Design patterns • Design principles • Complexity • Typology • Classification

Nomenclature

Classification, *typology*, and *taxonomy* are three terms used throughout this chapter. These terms are commonly used interchangeably. This chapter, however, will use Marradi's definitions [1]. *Classification* is the generic term for sorting a set by some defined metric, either quantitative or qualitative. *Systems of classification* or merely *classification* will be used as a generic term, encompassing both typology and taxonomy, as well as any other form of classification. *Typology* refers to a system of classification in which multiple means of division are simultaneously applied. As an example of this, when discussing government systems, a *theocratic, authoritarian government* has two means of classification applied and is the same as an *authoritarian, theocratic government*. A *taxonomy*, on the other hand, arises when multiple means of division are consecutively applied. The modern system

J.B. Reid • D.H. Rhodes (✉)
Systems Engineering Advancement Research Initiative, Massachusetts Institute of Technology, E17-361, 77 Mass. Ave, Cambridge, MA 02139, USA
e-mail: rhodes@mit.edu

A.M. Madni et al. (eds.), *Disciplinary Convergence in Systems Engineering Research*, https://doi.org/10.1007/978-3-319-62217-0_50

of sorting living species, the evolutionary taxonomy, is, as the name suggests, a taxonomy. Thus, *Homo sapiens* is not the same as a *sapiens Homo*.

50.1 Introduction

The word *emergence* as used by systems engineers and scientists can be vague, likely because it is intended to refer to a wide variety of behaviors that are uncommon, hard to predict, often lack defined structure. Nonetheless, these behaviors do seem to share some intuitive similarities that tease at potential generalizable theories regarding them. Many have attempted to more clearly define what emergence should mean, as to enable productive investigation into them and allow those generalized theories to be generated, tested, and proven. Due to the disparate forms of emergence expressed, a natural way to approach this is not to deal with emergence as a whole, but to break it down into categories that can be more readily conceptualized and investigated. Thus, classifications of emergence can be found in virtually any discussion of emergent phenomena. This chapter surveys several of these methods of classification, discusses the different problems they seek to address, and proposes the development of a set of emergent behavior design patterns based upon a certain form of typology. The expectation is that these design patterns will enable practicing systems engineers and operators of systems to anticipate potential emergent behavior, more quickly identify it, and have ready access to tools to deal with it. Additionally, these patterns have potential use for education new entrants to the field.

Emergence in this chapter is not synonymous with complex systems. The latter typically describes behavior, while the former more typically (though not always) describes some formal aspect of the system. Clearly, the two are closely linked with many complex systems exhibiting emergent behavior. Complex adaptive systems, for example, are noted for performing a certain emergent form of adaptation (as the name suggests) [2]. Due to this closeness, as well as the fact that neither emergence nor complex systems have definitive, universally agreed upon definitions, this chapter discusses many of these various definitions and means of classification, some overlap may occur.

50.2 Typologies of Emergence

Many past classifications of emergence have been taxonomies based on the degree of complexity, reducibility, or predictability of the system or behavior. Philosophers in particular tend to favor the latter two measures when constructing taxonomies. Reducibility (i.e., whether system-level emergent behavior is merely the result of complicated interactions of components or is fundamentally *other* from the components) is commonly used to differentiate *nominal* and *strong* emergence in an ontological sense (here ontological refers to the fundamental nature of what emergence is, regardless of whether or not we can ever fully understand it).

Nominal emergence is any form of system property that results from the combined properties of its components [3]. This includes things as simple as the fact that a watch can tell time due to its combination of gears and springs, all the way up to things as complicated as financial markets.

Strong emergence on the other hand refers to a behavior exhibiting "irreducible causal powers," [3] that is, it is an entity in its own right, not explainable in terms of its components, though it may still be dependent on them. The distinction here can be subtle and is not especially relevant here and thus will not be discussed further. Barnes [4] has an excellent explanation for interested readers. It should be noted that while some philosophers argue that conscious thought or life fits into the *strong* emergence category, scientists typically (but not universally) hold that the category is an empty set, as any case of *strong* emergence would be unable to be explained or replicated by the modern scientific method.

Predictability, which is used by philosophers to refer to what is theoretically predictable rather than what is practically or currently predictable, is often used to subdivide *nominal* emergence into further categories like *weak* emergence in an answer to an epistemological question (here epistemological refers to what can be known about emergence, regardless of what it genuinely is). *Weak* emergence is system-level behavior that, while caused by properties and interactions of its components (thus making it fit into the *nominal* category), is not easily explainable by them and requires simulation of potentially extremely high complexity to be predicted [3, 5].

Engineers typically are not overly interested in the ontological question of emergence and often dispense with reducibility as a metric. Additionally, rather than consider predictability in terms of the theoretical (e.g., you have unlimited computing power and infinite computational time), it is instead preferred to think of predictability in a more practical or current sense. When Bjelkemyr et al. use the *weak* and *strong* categories, for example, *weak* emergence is defined as that "which can be predicted by experience or extensive modeling and simulation," while *strong* emergence describes properties and behaviors that are displayed by the system, but cannot be reduced to known interactions or components, which cannot be attributed to an isolated subsystem or part [6]. This shift means that as our scientific understanding and modeling capabilities increase, a behavior could shift from one category to another, unlike in the philosophical typology. Additionally, this typology implies that there is no behavior that engineers and scientists cannot tackle.

Having only two categories in a taxonomy is somewhat limiting, however, and thus Maier expanded these two categories into *simple, weak, strong,* and *spooky* emergence, while making the emphasis on simulation and prediction even more explicit [7]. Basic systems, like mechanical watches, were downgraded to the *simple* category. The *weak* category was reserved for behavior that is consistently predictable using simulations of equal complexity to the actual system, but not with more abstract models. Most systems studied using agent-based models would fall into this category. *Strong* emergence is now a behavior whose general causal chain can be traced down to the component level, but simulations fail to consistently replicate. An excellent example of this is financial bubbles, which are easily explained on evening news, but resist accurate real-world predictions and diagnoses. Lastly, *spooky* emergence refers to all system-level behavior that is flatly inexplicable and unpredictable using current scientific knowledge, such as conscious thought.

Note that as these taxonomies have progressed, the emphasis on modeling and prediction has grown in importance. Some go as far as to say that emergence should be defined subjectively as whatever behavior is unexpected to an observer informed of the basic rules of the system [8, 9]. This makes the category that a behavior fits into not merely dependent on current scientific knowledge and engineering methodologies, but on the individual observer of the system. Under such definitions, it may occur that, for example, a specific behavior would not be emergent to an experienced system designer, but would be emergent to a novice system operator. This concept is potentially problematic for discussing emergence in a productive manner as certain complexity scientists have noted [2].

Not all engineering-specific systems of classification have focused on predictability, however. Complexity, commonly measured using amount of interaction with extra weight going to unique interactions, is another popular metric for distinguishing categories of emergence. Bar-Yam of the New England Complex Systems Institute based his emergence taxonomy on degree of interactions both among the components of the system, and between the system and its operating environment [10]. Szabo et al. even took the bold step of proposing a quantitative measure of emergence based upon interaction and possible system states [11]. Others have used grammar systems to formally define and quantify emergence in terms of interaction [12, 13]. Bar-Yam argued that these forms of classification enable a better understanding of the possible states of a system and more easily identify behavior dependencies. A robust, formal mathematical definition of emergence would certainly lend toward more accurately determining what aspects of systems lead to emergent behaviors. On the other hand, such a definition may exclude behaviors commonly considered to be emergent and have, as of yet, only been demonstrated on either rather basic systems or systems with few (or one) unique components. Thus, the feasibility of applying these definitions to more complicated designed systems, much less systems of systems, remains unproven. Note that these taxonomies should not be confused with similar ways of classifying complex systems by degree of complexity, such as Sheard's typology [14]. While such systems of classification are quite useful, they focus on describing the system as a whole, rather than specific behaviors that is the subject of this research.

The systems of classification (all taxonomies) discussed so far have many uses. For one, they serve to clarify what is meant by "emergent behavior" so that it can have genuine meaning in the systems engineering context other than "I didn't see that coming" (excluding the aforementioned exception to this). Additionally, they help inform design themes for handling emergence, such as robustness, communication, and learning [15], for defining risk of undesirable emergence [16], and general mindset that a systems engineer should have when confronting emergence

Table 50.1 Fromm's taxonomy of emergence [18]

Type	Subtypes	Examples
Type I: Simple/nominal emergence without top-down feedback	(a) Simple intentional (b) Simple unintentional	(a) Mechanical watch, steam engine (a) Pressure, temperature, slope of a sand-pile
Type II: Weak emergence including top-down feedback	(a) Stable (b) Instable	(a) Flocking, self-organization of the internet (b) Financial bubbles, demographic clustering
Type III: Multiple emergence with many feedbacks	(a) Stripes, spots, bubbling (b) Tunneling, adaptive emergence	(a) Zebra stripe formation, financial market cycles (b) Natural evolution, adaptive systems, scientific revolutions
Type IV: Strong emergence	None	(a) Life, culture

[17]. That said, these taxonomies are aimed more toward the experienced systems engineer and the development of theory. They do not lend well toward concretely explaining the concept of emergence to a layperson, nor do they immediately suggest concrete ways of either reducing the likelihood of a specific undesirable emergent behavior occurring or diagnosing and addressing such behavior once it has started occurring. Other forms of classification, based more on the type of expression of the emergent behavior, can better serve this purpose.

Fromm [18], for example, not only developed a taxonomy that still uses the basic simple-to-strong structure proposed by Bjelkemyr et al. [6] and Maier [7], but also added subcategories based on traits of the behavior exhibited, as can be seen in Table 50.1. Fromm's taxonomy is quite useful for illustrating emergence with real-world examples and for discussing potential causes (he does so mostly in terms of feedback loops across different spatial and temporal scales).

Mogul chose to dispose of the complexity hierarchy taxonomy altogether, instead creating a typology using descriptions of behavioral patterns, such as "unwanted synchronization" or "livelock," to create categories. Additionally, Mogul used the same structure to make a typology of causes of emergent behavior. The typology of behaviors can be seen in Table 50.2 *and the causes in* Table 50.3. It should be noted that Mogul did not intend this list to be exhaustive, but rather to serve as a starting point to be developed further [19]. While Mogul's typologies are restricted to undesirable emergent behavior (which he called "emergent misbehavior"), they could readily be expanded to include desirable behavior as well. Additionally, this "emergent misbehavior" of Mogul's appears to be a subset of Troncale's "system pathologies." [20].

Unlike many of the other systems of classification that were developed as taxonomies, Mogul's typology does not seek to place an individual instance of emergent behavior into certain type to the exclusion of others. Rather multiple behavior types may be exhibited in one instance of emergent phenomena. While the

Table 50.2 Mogul's proposed typology of emergent behavior [19]

Behavior types	Description	Examples
Thrashing	When competition over a resource results in switching costs between the sharing parties dominate actual useful work	Computers when insufficient working memory is allocated for the task at hand, a notable instance of this was the IBM/370 mainframe computers [21]
Synchronization	When time-varying behavior of components normally uncorrelated become correlated.	The frequency of pedestrians steps on the London Millennium Footbridge [22], router protocol message synchronization [23]
Oscillation	When the system periodically switches between states due to some feedback loop.	Gliders in Conway's Game of Life, PsEPR herding [19]
Deadlock	When progress stalls due to a circular set of dependencies	Mars Pathfinder software failure [24]
Livelock	When progress stalls because multiple components keep adjusting in such a way as to remain counteracting one another	Two individuals "dancing" in a hallway while trying to get around one another, Ethernet capture effect [25]
Phase change	When the system switches between states in an erratic or irreversible manner	Northeast Blackout of 2003 [26, 27], transition from synchronized traffic flow to a traffic jam [28]

Table 50.3 Mogul's proposed typology of causes of emergent behavior [19]

Type of cause	Description
Unexpected Resource sharing	The system designer assumed that separate components had access to separate resources when in fact the resources are shared and insufficient
Massive scale	The number of communicating components in the system is large enough to give rise to complex global behavior, even if individual components have simple behaviors
Decentralized control	Distributed systems that lack central controls, and hence suffer from incomplete knowledge and delayed information, can exhibit oscillations and chaos
Lack of composability	When components are not composable (i.e., lack strictly defined interfaces), modularity may not simplify behavior
Misconfiguration	Beyond literal errors, in complex systems, it is often too difficult for operators to understand global consequences of local configuration choices
Unexpected inputs or loads	While not all undesired behavior in response to an unexpected input or load is emergent, sometimes implementers draw the boundaries of "the system as a whole" too close to what they are responsible for implementing.
Information sharing delays	Latency makes a system harder to understand and harder to control, which can lead to oscillations and chaos.

matched frequency of pedestrian's steps on the London Millennium Footbridge is listed as an example of synchronization in Table 50.2, the overall behavior can be characterized as an oscillation caused by an expected feedback loop. Applying these "tags" on cases of emergent behavior gives system designers and operators another handle on which to conceptualize their own work, by enabling quick comparisons between similar cases to be made.

50.3 Design Patterns

This research seeks to generate and compile a set of design patterns to enable systems engineers to quickly recognize emergent behavior in their system, find similar cases in other systems, and identify possible avenues to address the mitigation or inducement of such behavior in the future. In this research, *design patterns* are used to refer to a subset of *design principles*. The latter refers to an abstract rule that produces concrete solutions to design problems. The *design patterns* are a subset of these aimed at specific problems, apply to the design itself (as opposed to the design process), and have some degree of provability or verifiability. Design patterns can be contrasted with *design heuristics*, the other subset of design principles, which are general, unverifiable rules-of-thumb largely aimed at the design process. Design patterns and heuristics have previously been used by Mekdeci [26] as well as Sheard and Mostashari [29] for similar purposes in the field of survivability and resilience. An example of a design pattern that arose based off the design principle "Margin" in Mekdeci's work is shown in Table 50.4. Though design patterns have been developed for numerous fields and predate the cited researchers [30], this chapter uses an altered version of Mekdeci's general format for describing them.

When formulating these design patterns, there are certain caveats that should be kept in mind. First, though design patterns are commonly referred to as general solutions to common problems based on a certain design principle, we do not claim

Table 50.4 Example design pattern for margin from Mekdeci [26]

Name	Margin
Type	Operational
Design principle	Margin [31, 32]
Problem	Capability is not being provided adequately
Context	Perturbation causes the output of system to be inadequate
Solution	Have system be capable of producing more output than necessary
Unintended consequences	May add require additional components, meaning unnecessary size (reducing the strategy of miniaturization), and/or unnecessary weight (making the strategy to mobility difficult to achieve)
Example	Extra-long wings on an A-10 aircraft, which generates enough lift in the event that part of the wings are destroyed or damaged
Related patterns	Heterogeneity, physical Redundancy

that these patterns "solve" emergence. While we disagree with the definition of emergence as the unexpected as some others have done [8, 9], most, if not all, forms of relevant emergence cannot be consistently predicted or designed for. Rather, through observing certain similarities in circumstances surrounding emergent behavior and the forms that certain behaviors take, we aim to expand the ability to at least anticipate it to some degree. So in this context, the phrases "suggested action to alleviate" and "suggested action to encourage" are used instead of "solution."

Similarly, while Mogul referred to his latter typology as a "taxonomy of causes" [19], we argue that the explicit cataloging of strict causal factors of emergent behavior is overly ambitious for what systems scientists are currently capable of (or may ever be capable of). Instead this research seeks to catalog common circumstances that are conducive to emergent behavior. For instance, a system merely being of massive scale (one of Mogul's listed causes) does not guarantee emergent behavior, nor can it often be truly stated as *the cause of* the emergent behavior. Rather, systems of massive scale tend to *be more likely* to exhibit emergent behavior, and accordingly we use the phrase "causal factor" to describe these traits. We define this in the following way: *A causal factor is any aspect of the system, which, when removed or changed, is likely to reduce the occurrence of emergent behavior, or, when induced, is likely to increase the occurrence of emergent behavior*. Clearly, this limits the testability of the design patterns. Nonetheless, it is expected that many of these patterns will be noncontroversial for experienced systems engineers and appear reasonable upon reflection. Moreover, we believe that these patterns will prove useful to practicing systems engineers, particularly those with limited experience with emergent phenomena.

The first step toward developing the proposed design patterns is to generate a typology of emergent behavior, a set of common causal factors, and suggested actions to either alleviate or induce the behavior. The suggested actions will form the basis of each design pattern. The required pieces of information should not be listed independently, but instead linked to one another along with concrete examples.

Table 50.6 shows a summary of such information for three types of emergent behavior – synchronization, phase change, and oscillation. While this table contains descriptions and potential consequences of each emergent behavior, it lacks clear explanations of the common causal factors and the suggested actions. Some of the former are more clearly defined in Table 50.5, while the latter will be more fully described in the proposed design patterns, discussed later in this chapter. A key element is to keep each type well documented both in its examples and in its suggested actions. For example, "over design capacity" has been found to be a feasible action in a variety of domains: from traffic jams, where the transition from synchronized flow to congestion cannot occur over a certain density [33, 34], to online communication, well illustrated by the PsEPR herding case where the initial herding behavior and oscillation from one server to the next cannot occur unless the number of clients is above a certain threshold [19]. Note that the common causal factor "density" exists in both of these examples as well. These can be found both in

Table 50.5 Descriptions of three types of causal factors

Causal factor	Explanation	Examples
Density	As the energy or material density of a system increases, feedback loops are accelerated and strengthened, nonlinear behavior becomes more apparent, and likelihood of a perturbation increases.	Spontaneous traffic jam [28]; laminar vs. turbulent fluid flow; locust swarms [37]; PsEPR herding [19]
Frequency matching	Two or more known feedback loops or oscillations that were expected to have different frequencies are instead found to have mutually reinforcing frequencies.	Vortex-induced vibration, London Millennium Footbridge vibration [38]
Perturbation/ nucleation site	There exists some temporally and spatially limited variation to input to the system, either endogenous or exogenous, that can agitate local non-linearities, potentially initiating a chain reaction across the system	Laminar vs. turbulent fluid flow; spontaneous traffic jam [28]

field-specific research (as the above examples were) or in more generalized discussions of emergent behavior, such as De Wolf's and Holvoet's proposal of two specific design patterns for dealing with emergence resulting from decentralized coordination: gradient fields and market-based control [35]. This general system of cataloging is not without precedence and has been used previously to link cause, effect, solution, and examples for different types of perturbations [36] (Table 50.6).

Once the above catalog has been created (or at least a substantial start has been made), it is then possible to use the suggested actions as design patterns. Table 50.7 shows an example of such a design pattern for "over design capacity." In a full digital catalog, the various causal factors, suggested actions, and examples could be linked to the expanded explanations to facilitate easy exploration by the reader. Similarly, the related patterns could also be linked.

While this format is useful for a reader to quickly reference information about specific behaviors, it is limited in its ability to search in the opposite direction (e.g., a designer has noticed that their system-as-designed contains some unexpected feedback loops and want to understand what kind of behaviors might occur if this not addressed before implementation). To serve this need, a cross-mapping table, relating causal factors and suggested actions to emergent behavior, has been generated and is shown in Table 50.8.

Neither the design pattern nor the cross-mapping table is intended to be exhaustive. Instead, they are intended to serve as a structure for evolving a full catalog.

There are four primary intended users of these design patterns: systems architects/engineers (the people designing and creating the system); system operators (the people running the system on a day-to-day basis); analysts (those charged with diagnosing problems and rectifying them); and students. Designers can use these

Table 50.6 Descriptions, causal factors, and suggested actions for three types of emergent behavior

Emergent behavior	Synchronization	Phase change	Oscillation
Description of emergent behavior	Components of the system or of the environment whose time-varying behavior is expected to be uncorrelated become correlated	The system switches between two or more states in an irregular, unintended, or irreversible way	The system or some set of components periodically switch back and forth between two or more states in some repeating pattern.
Consequences	Can increase the load on a component or the system as a whole, alternately may result in greater efficiency of a system that is normally random	Can cause the system to act unpredictably, making control and monitoring more difficult. Can also dramatically increase or decrease system performance.	The rapid state changes can inhibit productive function or cause undue loading to some parts of the system.
Common causal factors	Unexpected inputs; Decentralized control; frequency matching; unexpected feedback loops, density	Density; perturbation/ nucleation site	Unexpected feedback loops, frequency matching
Suggested action(s) to alleviate	Over design capacity; induce randomness; introduce perturbations	Remove perturbations; over design for capacity	Decoupling; induce randomness
Suggested action(s) to encourage	Minimize perturbations; market-based control; gradient fields	Induce perturbations; add energy	Introduce time delays
Example	Frequency of pedestrian steps on the Millennium Footbridge [22]; router protocol message synchronization [23]; PsEPR herding [19]; Laminar fluid flow	Spontaneous traffic jam [28]; laminar vs. turbulent fluid flow; cascading network failure [26, 27]; locust swarms [37]	Vortex-induced vibration, London Millennium Footbridge vibration [38]; PsEPR herding [19]; gliders in Conway's Game of Life

patterns to help anticipate emergent behavior and adjust their plans to either encourage or discourage the development of emergent behavior once the system is put into operation.

Both operators and analysts can use these patterns to more quickly identify and react to emergent behavior in their systems. This is line with the Cynefin framework's suggest course of action "probe-sense-respond" in the complex domain [40]. By showing system operators and analysts the common causal factors and common suggested actions to alleviate, they more quickly identify the correct elements of their system to probe, know what they should be looking to sense, and develop a response. As discussed previously, the historical taxonomies have focused on the systems architect's role in working with emergence and been less relevant to the analyst. Previously, this need has been noted by De Wolf and

Table 50.7 Example proposed design pattern

Over design capacity/Increase capacity	
Emergent behavior types addressed	Synchronization; phase change
Context	Emergent behavior, potentially exacerbated by the material or energy density in the system, results in higher than expected load on some component or the system as a whole
Action	Allow for extra load capacity than would be necessary should no emergent behavior occur.
Unintended consequences	May result in induced demand; will likely require additional resources (particularly physical space) that may be costly or not be available
Example	Increased memory capacity resulted in trashing behavior like that seen in the IBM/370 become less common [21]; opening the shoulder of a highway as an additional lane can (at least temporarily) reduce traffic [39]
Related patterns	Margin, heterogeneity, physical redundancy

Holvoet, who attempted to remedy it by applying design patterns to a subset of emergent behaviors, specifically decentralized coordination [35].

Lastly, "students," be they grade school or university students or practicing engineers learning about systems engineering, can use these design patterns to get a more concrete grasp on what emergence means in applied use. Given emergence has highly varied meanings across fields (and even within them) as well as its common English definition of "becoming visible over time," the concept can be difficult to teach to students. This is particularly true for those who are not native English speakers as it requires a nonliteral interpretation of the colloquial meaning of "emergence." An anecdotal experience suggests this difficulty in explanation is eased through the use of examples from a variety of domains, particularly those with personal or professional familiarity to the student. The proposed catalog of design patterns would formalize this process and allow for more detail and clarity in the explanation. Once a student has been familiarized with these various types of emergent behavior and examples of them, they will be better prepared to consider the systems of classification proposed by Bar-Yam [10], Maier [7], and the others, as well as the more general modes of thought useful for tackling emergence, such as those discussed by Keating [15].

While most of the examples of emergent behavior cited in this chapter are those of designed systems, the natural world exhibits innumerable instances of emergent behavior that is also worth considering. Such cases have been thoroughly studied by organizations like the Sante Fe Institute and often demonstrate aspects of direct relevance to system designers, such as evolved modularity [41]. The effort to develop this design pattern catalog is in early stages. Ongoing research continues to investigate additional categories of emergent behavior, as well as identifying case examples of such behavior.

Table 50.8 Emergent behavior cross-mapping

		Common causal factors						
		Density	Perturbation/ nucleation site	Unexpected feedback loops	Frequency matching	Unexpected inputs	Decentralized control	Massive scale
*Su*ggested actions to alleviate	Over design capacity (margin)	Synchronization		Synchronization	Synchronization	Synchronization	Synchronization	
	Induce randomness	Synchronization		Synchronization Oscillation	Synchronization Oscillation	Synchronization	Synchronization	
	Induce perturbations	Synchronization Phase change	Phase change	Synchronization Oscillation	Synchronization Oscillation	Synchronization	Synchronization	
	Decoupling				Oscillation			
	Remove perturbations	Phase change	Phase change					

Acknowledgments The authors gratefully acknowledge funding for this research provided through MIT Systems Engineering Advancement Research Initiative (SEAri, http://seari.mit.edu) and its sponsors. We would also like to thank former SEAri research assistant Lucie Reymondet for her initial investigation in this research effort.

References

1. Marradi A (1990) Classification, typology, and taxonomy. Qual Quant 24(2):129–157
2. Holland JH (2002) Complex adaptive systems and spontaneous emergence. In: Curzio AQ, Fortiz M (eds) Complexity and industrial clusters. Physica-Verlag, New York, pp 25–34
3. Bedau M (2002) Downward causation and the autonomy of weak emergence. Principia 6 (1):5–50
4. Barnes E (2012) Emergence and fundamentality. Mind 121(484):873–901
5. Seager W (2005) Emergence and efficiency. In: Erneling C, Johnson D (eds) The mind as a scientific object, 1st edn. Oxford University Press, New York, pp 176–190
6. Bjelkemyr M, Semere D, Lindberg B (2008) Definition, classification, and methodological issues of system of systems. In: System of systems engineering—principles and applications. Taylor Francis CRC Publishers, Boca Raton
7. Maier MW (2015) The role of modeling and simulation in system of systems development. In: Rainey LB, Tolk A (eds) Modeling and simulation support for system of systems engineering applications. Wiley, Hoboken, pp 11–44
8. Ronald EMA, Sipper M, Capcar MS (1999) Design, observation, surprise! A test of emergence. Artif Life 5(3):225–239
9. White BE (2007) On interpreting scale (or view) and emergence in complex systems engineering. In: 1st annual IEEE systems conference, Honolulu, p 1–7
10. Bar-Yam Y (2004) A mathematical theory of strong emergence using multiscale variety. Complexity 9(6):15–24
11. Szabo C, Teo YM, Chengleput GK (2014) Understanding complex systems: using interaction as a measure of emergence. In: Proceedings of the 2014 winter simulation conference, Savannah, p 207–218
12. Kubík A (2003) Toward a formalization of emergence. Artif Life 9:41–65
13. Baas NA, Emmeche C (1997) On emergence and explanation. Intellectica 25(2):67–83
14. Sheard S (2010) A complexity typology for systems engineering. In: Twentieth annual international symposium of the international council on systems engineering, Chicago
15. Keating CB (2009) Emergence in system of systems. In: Jamshidi M (ed) System of systems engineering: innovations for the 21st century. John Wiley & Sons Inc., Hoboken, pp 169–190
16. Ferreira S, Faezipour M, Corley HW (2013) Defining and addressing the risk of undesirable emergent properties. In: IEEE international systems conference, Orlando
17. Mcconnell GR (2008) Emergence : still a challenge for the systematic. In: INCOSE international symposium, Utrecht, p 720–734
18. Fromm J (2005) Types and forms of emergence. arXiv preprint nlin/0506028
19. Mogul JC (2006) Emergent (mis)behavior vs. complex software systems. Proceedings of the 2006 EuroSys conference on - EuroSys ‘, Leuven, 40(4)293
20. Troncale L (2011) Would a rigorous knowledge base in systems pathology add significantly to the systems engineering portfolio? In: Proceedings of the 9th conference on systems engineering research. Los Angeles
21. Denning PJ (1968) Thrashing: its causes and prevention. In: Proceedings of the fall AFIPS joint computer conferences, San Francisco, p 915–922
22. Eckhardt B, Ott E, Strogatz SH, Abrams DM, Mcrobie A (2007) Modeling walker synchronization on the millennium bridge. Phys Rev E 75:021110

23. Floyd S, Jacobson V (1994) The synchronization of periodic routing messages. IEEE/ACM Trans Networking 2(2):122–136
24. Reeves GE (1997) “What really happened on mars? The Authoritative Account,” Microsoft Research, [Online]. Available: http://research.microsoft.com/en-us/um/people/mbj/Mars_Pathfinder/Authoritative_Account.html
25. Ramakrishnan KK, Yang H (1994) The ethernet capture effect: analysis and solution. In: Local computer networks, 1994. Proceedings., 19th conference on, Minneapolis. p 228–240
26. Mekdeci B (2013) Managing the impact of change through survivability and pliability to achieve viable systems of systems. Massachusetts Institute of Technology, doctoral thesis, Cambridge, MA
27. Glass RJ, Beyeler WE, Stamber KL (2004) Advanced simulation for analysis of critical infrastructure: abstract cascades, the electric power grid, and fedwire 1 advanced simulation for analysis of critical infrastructure 1, (No. SAND2004-4239). Sandia National Laboratories, Albuquerque
28. Helbing D (2001) Traffic and related self-driven many-particle systems. Rev Mod Phys 73 (4):1067–1141
29. Sheard S, Mostashari A (2008) A framework for system resilience discussions. In: INCOSE international symposium, Utrecht, p 1243–1257
30. Alexander C, Ishikawa S, Silverstein M (1977) A pattern langugage: towns, buildings, construction. Oxford University Press, Oxford, UK
31. Jackson S, Ferris TL (2013) Resilience principles for engineered systems. Systems Engineering 16(2):152–164
32. Richards, M. G. (2009). Multi-attribute tradespace exploration for survivability. PhD Thesis, MIT, Cambridge, MA
33. Tadaki S-I et al (2014) Critical density of experimental traffic jam. In: Chraibi M, Boltes M, Schadschneider A, Seyfried A (eds) Traffic and granular flow ‘13. Springer International Publishing, New York City, pp 505–511
34. Kerner BS, Konhauser P (1993) Cluster effect in initially homogeneous traffic flow. Phys Rev E 48(4):2335–2338
35. De Wolf T, Holvoet T (2007) Design patterns for decentralised coordination in self-organising emergent systems. In: Brueckner SA, Hassas S, Jelasity M, Yamins D (eds) Engineering self-organising systems. Springer, Berlin, pp 28–49
36. Mekdeci B, Ross AM, Rhodes DH, Hastings DE (2012) A taxonomy of perturbations: determining the ways that systems lose value. In: 2012 I.E. international systems conference, Proceedings. Vancouver, Canada, pp 507–512
37. Buhl J et al (2006) From disorder to order in marching locusts. Science 312(5778):1402–1046
38. Dallard P et al (2001) The London millennium footbridge. Struct Eng 79(22):17–33
39. Formby B (2016) Once TxDOT opened up SH 161 shoulders, traffic started sailing. The dallas morning news, Dallas
40. Kurtz CF, Snowden DJ (2003) The new dynamics of strategy: sense-making in a complex-complicated world. IBM Syst J 42(3):462–483
41. Solé RV, Valverde S 2007 Spontaneous emergence of modularity in cellular networks.,” Sante Fe, 2007-06–013

Chapter 51
Collective Behaviors: Systemic View of Distinct Forces in a New Framework

Arash Vesaghi, Nasrin Khansari, and Mo Mansouri

Abstract Investigating emergence properties of complex adaptive systems has received a great deal of attention since Schelling's model of segregation and Granovetter's threshold model on opinion dynamics. In the past 50 years, the notion of consensus as emerging characteristic has been studied in different disciplines such as sociology, psychology, economics, systems engineering, and physics.

Different models have been proposed to explain and examine the collective behaviors. The topic is mostly discussed as social phenomenon, and opinion dynamic models are the foundation to explain consensus. Thus, identifying underlying mechanisms is a common concern. Despite extensive literature, absence of unified framework to consolidate these studies is clear. Here, a short review of existing literature and proposed framework is presented to assist with binding these studies. The proposed framework classifies three independent forces: (1) External, (2) Network, and (3) Memory. Measuring influence of these forces in some cases is hard; nevertheless, the appreciation of distinct processes is valuable for decision-makers. Each force should be activated or countered accordingly.

Keywords Complex adaptive systems • Collective behavior • Decision-making • Network effects • Network externalities

51.1 Introduction

When it comes to some phenomenon, there are different terminologies to describe the same process with respect to the discipline of researchers. For instance, herding in sociology [1] or lock-in process [2] in economics are different terms used to describe the collective behavior. In general, disagreements and chaos are expected

A. Vesaghi (✉) • M. Mansouri
School of Systems and Enterprises, Stevens Institute of Technology, Hoboken, NJ 07030, USA
e-mail: avesaghi@stevens.edu; mo.mansouri@stevens.edu

N. Khansari
Electrical & Systems Engineering Department, University of Pennsylvania, Philadelphia, PA, USA
e-mail: khansari@seas.upenn.edu

A.M. Madni et al. (eds.), *Disciplinary Convergence in Systems Engineering Research*, https://doi.org/10.1007/978-3-319-62217-0_51

outcomes in a diverse system with multiple stakeholders. Collective behaviors are exceptional processes, which are result of consensus among individuals [3]. Understanding underlying forces in convergence processes is critical in defining the system and influencing its outcome.

Social scientists and psychologists have proposed several theories to explain collective behaviors in society. The basic premise of all theories is about how individuals are affecting each other's decisions [4]. Influencing others, whether through personal relations or logical explanation, could change how choices are perceived and evaluated by individuals, especially if decisions have universal impacts [5]. The decision analysis theories are developed based on the premise of assigning a utility to each option and comparing options' utilities [6].

In addition to utility theory, other simpler models have been developed to investigate collective behaviors. The model for segregation developed by Schelling in 1971 is one of the premiers. The model is simple and elegant. Schelling used one parameter to describe the behavior (or action) of individuals. The model showed how independent actions by individuals resulted in a segregated community [7]. The assumption of this model was consistent with existing literature in social psychology about prejudice [8].

There are classic models developed to study opinions among individuals in society. One might argue having an opinion is not the same as performing an action. While this argument is valid to the extent of studying difference between an action and corresponding opinion (or position), it does not have any relevance to the topic of this chapter. Simply put, a decision and action caused by the decision are the same from the perspective of an observer. Granovetter, in 1978, has introduced a model to examine opinion dynamics in the society. The model, known as threshold model, describes a binary system. In this system, every individual has an option to participate or not. Granovetter discusses his results and explained difficulties involved in the model including measuring thresholds [9].

In the past two decades, because of significant increase in computational capacity, many researchers have developed more complex models to investigate complex adaptive systems. The term complex adaptive system was coined to describe a distinction between complex physical (natural) systems and adaptive systems. The adaptivity, as characteristic of a system, requires some level of intelligence, or consciousness [10]. Nevertheless, agent-based modeling (ABM) became dominant methodology in describing such systems [11]. ABM is based on the cellular automata concept, which Neuman described as a process to break a system into subsystems (individual parts) [12].

Recent studies using ABM have investigated different problems. Majority of these incredible works focused on investigating different individuals' behaviors and their correlation to equilibrium outcome or final state of the system under simulation [13]. When it comes to studying collective behaviors, there is a gap in literature regarding the underlying processes and their correlation to dynamics of the entire system. This chapter focuses on developing a platform which encompasses forces entangled in forming collective dynamics. The background literature is discussed in

Sects. 52.2 and 52.3. The proposed framework is introduced in Sect. 52.4. Finally, further works and conclusion are discussed in Sect. 52.5.

51.2 Decisions

The issue of individuals' decisions and its relation to expected (or actual) outcome is a frequent topic in various studies. The subject of decision-making does not apply to complex physical systems. Elements in complex physical systems are defined in a way that they lack ability to make a decision. Elements are bonded by laws of nature to function in exact ways. Studies regarding decision-making mostly focus on systems, which are made up of humans or human-like smart entities [14]. A decision could be simple (e.g., choosing an ice cream flavor) or complex (e.g., engaging in a war or not). A decision could be personal (e.g., proposing to a girl) or social (e.g., voting for a candidate). Nevertheless, the emphasis of this chapter is on decisions with collective aftereffects.

Decisions with collective effect are a specific class with specific characteristics [15]. In a system with diverse (possibly heterogenous) subsystems, chaos, disagreement, or disharmony are the expected outcomes. In other words, detecting continuous increase in entropy is expected in dynamic systems (e.g., society). Consequently, discovering patterns of collective behaviors requires further explanation and discussion.

It is necessary to examine existing literature on decision-making to illuminate the difference between decision classes. Psychology is the primary discipline in studying human behaviors. The American Psychology Association defines the term decision-making as "The process of choosing between alternatives; selecting or rejecting available options" [16]. Psychologists investigate people's behaviors and mindset to understand how people observe, evaluate, and decide. Besides studying mental disorders, the attention is regarding the decision-making process under the assumption that individuals are, at least partially, rational [17].

Kahneman and Tversky perform a series of experiments to study differences between heuristic and rational decision-making processes. Their findings show that heuristic-based decisions are not illogical or irrational [18]. Nevertheless, the argument gives rise to the discussion about decision-making processes. Recently, Kahneman and Thaler propose that utility maximization is not sufficient to explain certain behaviors [19]. The term *satisficing* best explains this fact. In addition, latest studies in neuroscience suggest a two-stage process in decision-making. Stage one is dedicated to evaluating options, and stage two is dedicated to finalize the action [20].

Economists have made a significant effort in transforming psychological and social experiments into mathematical models. The most recognized model is known as multicriteria decision-making. This method breaks down an option into certain features. Then, each individual inclination toward these features is added to evaluate the option. The linear form of this process could be expressed as the inner production of two vectors [21]. Equation 51.1 is an example of such a utility function:

$$U_i(t,x) = \vec{f_x}(t).\vec{a_i}(t) \tag{51.1}$$

In this method, there is no restriction over changing the feature variables or individuals' inclinations over time. There are different procedures to implement such changes in a system. The subject of inclination changes could best be noticed in social learning models. The topic of learning process, either as a social process or algorithm-based, is widely covered by many researchers [22, 23]. The concept of learning offers a path to adaptivity in complex adaptive systems. In addition, the feature (or characteristic) of options could evolve over time as a result of aggregated decisions.

51.3 Modeling and Simulation Background

There are different tools used to model complex systems. Mathematicians and physicists developed majority of these tools, i.e., statistical physics is one example. Consequently, all these methods are developed for less diverse (homogenous) systems. Even under extreme computer-based simulations, subsystems behave in a predictable manner. The bridge between existing models and simulation methods is the concept of decision-making [24]. The decision-making rules are laws of the nature for human or human-like entity.

Introducing the notion of decision-making into classical models resulted in new methods and further developments. One of the great examples is the game theory. Game theory is a branch of mathematics dealing with optimizing outcome of designed (abstract) games. Game theory becomes an important economic analysis tool after Nash proves the existence of an equilibrium in noncooperative games. Nash proves the existence of at least one mixed strategy equilibrium in noncooperative games [25]. Consequently, game theory is broadly used to investigate different equilibrium conditions in mutliplayer games. Game theory, despite its achievement to explain individuals' behaviors, is not sufficient to elucidate collective behaviors [26]. All collective behaviors are not caused by some type of optimization in a system.

The other methodology used in modeling complex adaptive systems is statistical and probability analysis. Physicists developed several models to solve various many-body problems. This type of analysis is complemented by ordinary differential equation (ODE) models. The notion of using ODE to study complex adaptive systems was founded by Forrester as part of his effort to advance system dynamics [27]. Consequently, these models are defined to project the trends of expected values. For example, SIR model was developed to predict (or at least explain) the trajectory of epidemics. These models are quite useful in policy analysis

[28]. Nonetheless, ODEs and statistical inferences are not capable to elucidate collective behaviors. For instance, introducing networks in a model would result in nonlinearity, and statistical methods are not capable of solving such problems. There are several approximation methods, such as mean field theory, which assist with solving some problems. However, too much simplification gives rise to inaccuracy.

The final methodology is the agent-based modeling (and simulation) or ABM. ABM is a methodology developed as a descendant of cellular automata. ABM is based on describing an environment and group of agents. It is a discrete approach. This methodology is effective in analyzing different problems. It is broadly used in economics, sociology, and systems engineering [29]. It provides new solutions for some of the old problems. For example, in the case of SIR model, an alternative model was developed recently to compare the result of ABMs and SIR model [30]. This example shows how this methodology could resolve some of the issues with present solutions. This methodology is suitable for investigating collective behaviors, as it was shown with several examples at the beginning of the chapter. Thus, this approach is used in the proposed framework.

51.4 Proposed Framework

The framework is developed to unify current models and descriptions on the topic of collective behaviors. The model assumes that any available option is offered by a supplier, and it is either selected or rejected by individuals. Supply side of such a transaction could include a politician offering a new policy, a company marketing a product, or an activist group promoting a position. The dynamics and activities on the supply side is not the topic of discussion in this chapter.

The framework is designed to investigate decisions, which have common aftereffect and are significant. Complicated personal decisions such as a decision to get married are not subject of this chapter, neither decisions without common aftereffect, even if those decisions are influenced by others' behaviors, are subject of this paper. In other words, decisions which have several criteria, and corresponding inclination factors, do not generate collective behaviors. This framework addresses issues regarding the convergence toward steady states and reaching consensus among individuals. The consensus among individuals results in collective behaviors.

There are three forces forming the collective behaviors. The first force is the perception of options' intrinsic values or utilities. This part is common with current literature on decision-making. The second force is the network effect, when others' actions influence individuals' decisions. The last force is personal experience or memory effect. The concept of personal experience is based on the notion of switching cost in economics, when a system has a memory. Since intrinsic values of options depend on options features and they are controlled (or at least assumed to be controlled) by supplier, then distinction between two forces is necessary. It is

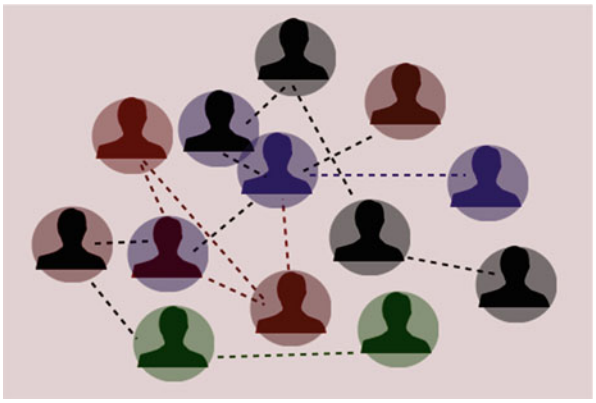


Fig. 51.1 Combination of three forces

notable that this concept is different from personal tendency toward an option. If a personal tendency exists, it will constantly increase the value of one option over others. In case of personal experience, the tendency toward each option could change. The memory is not fixed in advance.

As mentioned previously, options related to decisions with collective aftereffects are simple. Options with multiple features, which correspond to idiosyncratic inclinations eventuate to diversity without consensus. For example, there is no expectation of consensus in art or fashion. The number of significant features associated with the options, that are subject to collective behaviors, is limited to one or two. On the other hand, features of such options are either directly influenced by collective decisions, or they are completely independent of decisions. In case of being completely independent, their value is not important in predicting the trajectory of the system. In this case, if they are affected by collective decisions, then their evolution could be modeled similar to random reinforcement processes. One of great examples of such a process is Polya urn, which was used by Arthur in explaining path dependency [31]. Figure 51.1 illustrates combination of these forces. Color of each individual corresponds to their choice. Spheres around individuals correspond to their personal experience. Links between individuals correspond to network force, and overall color of the environment corresponds to dominating option in the system.

The second force is the network effect. This process is well known and broadly studied. However, in most studies the emphasis is limited to identifying the impact of network structure on equilibrium-state values. Some studies explored other avenues such as investigating social network structures [32] and, propagation on network [33]. Nevertheless, there are few investigations examining combination of forces on a system.

The last force is personal experience. The idea of switching cost in economics inspired the definition of this force. Put simply, in sequential decision-making processes, each new decision could rely on earlier decisions. For example, this force affected the progression of keyboards into the current model. Studies regarding the QWERTY investigated this force thoroughly [34].

Combining these forces into a single framework for approaching problems about collective behaviors assists with presenting better understanding of complex adaptive systems. While measuring impact of each force is hard as an empirical practice, observing secondary variables could be easier. In a recent publication, the authors offer a simpler model to examine the possibility of measuring how fast a system evolves from unstable equilibrium state into equilibrium state as a measurable variable [35]. Equation 51.2 summarizes these forces:

$$U_i(x,t) = \alpha V_{\text{int}}(x,t) + \beta V^i_{\text{net}}(x,t) + \gamma V^i_{\text{pers}}(x,t) \tag{51.2}$$

Furthermore, this framework is temporal. Consequently, force variables introduced in the framework could evolve over time. Even though fixing those factors will not change the trajectory of the system, it could change how fast or slow the system would evolve. This framework applies to constrained (or closed) systems, in which introduction of new options or evolution rule of options is known.

51.5 Conclusion

Collective behaviors are notable emergent properties of complex adaptive systems. In one perspective, collective behaviors are perceived as a process relating to each subsystem. In other perspective, collective behaviors are perceived as system-wide process generated by subsystems' decisions. While studying how individual subsystems are going to position themselves in the system is valuable, it lacks clarification on how the system is changing as a whole. In some cases, such as segregation, investigating a system as a unity offers valuable insight.

The proposed framework could be used in different disciplines and modified in line with specific problem. Various simpler types of this framework are being used in modeling several problems. In addition, the description of the framework could potentially help with improving appreciation of different forces and their effects on system path.

With respect to recent increase in computational capacity, running computer-based simulations to examine complex adaptive systems and their behaviors is obtainable more than ever. A proposed framework in this chapter helps with unifying models in different disciplines. Future work will focus on more detailed mathematical descriptions and computer simulations to examine practicality of measuring empirical variables in practice. Applying this model directly in many cases will be difficult. Instead, results of models built on top of this framework could be used directly in correspondence with empirical evidence. Authors are working on simulations to examine this framework with respect to some studied problems.

References

1. Raafat RM, Chater N, Frith C (2009) Herding in humans. Trends Cogn Sci 13:420–428
2. Witt U (1997) "lock-in" vs."critical masses"—industrial change under network externalities. Int J Ind Organ 15:753–773
3. Blumer H (1971) Social problems as collective behavior. Soc Probl 18:298–306
4. Dutton JE, Ashford SJ (1993) Selling issues to top management. Acad Manag Rev 18:397–428
5. Acemoglu D, Ozdaglar A (2011) Opinion dynamics and learning in social networks. Dyn Games Applic 1:3–49
6. Tversky A, Kahneman D (1992) Advances in prospect theory: cumulative representation of uncertainty. J Risk Uncertain 5:297–323
7. Schelling TC (1971) Dynamic models of segregation. J Math Sociol 1:143–186
8. Allport GW, Kramer BM (1946) Some roots of prejudice. J Psychol 22:9–39
9. Granovetter M (1978) Threshold models of collective behavior. Am J Soc 83.6(1978):1420–1443
10. Buckley W (1968) Society as a complex adaptive system. In: Modern systems research for the behavioral scientist. Aldine, Chicago
11. Batty M (2007) Cities and complexity: understanding cities with cellular automata, agent-based models, and fractals. The MIT press, Cambridge/London
12. Von Neumann J (1951) The general and logical theory of automata. Cereb Mech Behav 1:1–2
13. Darabi HR, Mansouri M (2013) The role of competition and collaboration in influencing the level of autonomy and belonging in system of systems. IEEE Syst J 7:520
14. Snowden DJ, Boone ME (2007) A leader's framework for decision making. Harv Bus Rev 85:68
15. Rossi F (2014) Collective decision making: a great opportunity for constraint reasoning. Constraints 19:186–194
16. Gerrig RJ, Zimbardo PG, Campbell AJ, Cumming SR, Wilkes FJ (2011) Psychology and life. Pearson Higher Education, AU
17. Arthur WB (1994) Inductive reasoning and bounded rationality. Am Econ Rev 84.2:406–411
18. Tversky A, Kahneman D (1974) Judgment under uncertainty: heuristics and biases. Science 185:1124–1131
19. Kahneman D, Thaler R (2006) Anomalies: utility maximization and experienced utility. J Econ Perspect 20:221–234
20. Glimcher PW, Fehr E (2013) Neuroeconomics: decision making and the brain. Academic Press, Boston
21. Berry ST (1994) Estimating discrete-choice models of product differentiation. The RAND J Econ :242–262
22. Bandura A, McClelland DC (1977) Social learning theory. Prentice Hall, Englewood Cliffs
23. Sutton RS (1988) Learning to predict by the methods of temporal differences. Mach Learn 3:9–44
24. Dooley KJ (1997) A complex adaptive systems model of organization change. Nonlinear Dynamics Psychol Life Sci 1:69–97
25. Nash J (1951) Non-cooperative games. Ann Math:286–295
26. Sally D (1995) Conversation and cooperation in social dilemmas a meta-analysis of experiments from 1958 to 1992. Ration Soc 7:58–92
27. Forrester JW (1970) Urban dynamics. IMR Ind Manag Rev (pre-1986) 11:67
28. Khansari N, Vesaghi A, Mansouri M, Mostashari A The multiagent analysis of social progress in energy behavior: the system dynamics methodology. IEEE Syst J:1
29. Tesfatsion L (2001) Introduction to the special issue on agent-based computational economics. J Econ Dyn Control 25:281–293
30. Rahmandad H, Sterman J (2008) Heterogeneity and network structure in the dynamics of diffusion: comparing agent-based and differential equation models. Manag Sci 54:998–1014

31. Arthur WB (1989) Competing technologies, increasing returns, and lock-in by historical events. Econ J 99(394):116–131
32. Barabási A-L, Albert R (1999) Emergence of scaling in random networks. Science 286:509–512
33. Watts DJ (2002) A simple model of global cascades on random networks. Proc Natl Acad Sci 99:5766–5771
34. David PA (1985) Clio and the economics of QWERTY. Am Econ Rev 75(2):332–337
35. Vesaghi A, Mansouri M (2016) Externalities and peer effects of collective adoption in networks. In: 2016 11th system of systems engineering conference (SoSE)

Chapter 52
Generational Evolution in Complex Engineered Systems

L. Dale Thomas and Katherine Burris

Abstract Systems need to evolve to what is required and needed functionally. Looking at how each generation of a system changes as an evolutionary process, one can see correlations with the natural evolution process. Using nature as a guide, biological models may be a means to assess an engineered system and propose anticipatory changes (evolutions) to a current system to create the next generation. This idea of systems generational evolution may provide insight into the design process of multigenerational systems to be more adaptable to changing technology. Fundamental areas of research for support in this topic are biological evolutionary mathematical models and system of systems classification.

Keywords Evolvability • Systems generational evolution • Complex engineered systems

52.1 Introduction

In January 2016, the Department of Defense announced the purchase of 32 C-130Js from Lockheed Martin. This announcement was noteworthy because it will likely make the C-130 "the first military aircraft in history to stay in continuous service for a hundred years" [1]. Just a few months prior, aircraft maker Sikorsky announced plans to turn a retired Black Hawk helicopter into an "optionally piloted" aircraft, with Army officials confirming that a recent test of the prototype proved to be a success [2]. On a smaller scale, successive generations of the venerable iPhone are rolled out by Apple every 2 years or so, each marking an evolutionary advance over the previous generation. The characteristic in common among a cargo aircraft, a helicopter, and a mobile phone is evolvability. However, evolvability is a concept much easier to grasp in abstract than to measure, much less to optimize. One thing that we can assert with respect to evolvable systems is that nature has a proven approach for the development of complex systems and perhaps can provide some useful insights for the development of complex engineered systems.

L. Dale Thomas (✉) • K. Burris
University of Alabama in Huntsville, Huntsville, AL, USA
e-mail: dale.thomas@uah.edu; keb0023@uah.edu

A.M. Madni et al. (eds.), *Disciplinary Convergence in Systems Engineering Research*, https://doi.org/10.1007/978-3-319-62217-0_52

52.2 Background

System evolution is the process by which a system physically transforms from one configuration to a more desirable configuration [3]. Ideally, the evolvability of a system design allows engineers to continually and easily discover and implement performance-improving (or value creation in general) variants of the system design [4]. Three drivers of evolutionary change in an engineered system include the emergence of new technology, a change in operational environments, or a change in stakeholder needs [5].

The need for evolution in complex engineered system (CES) design is well established in the literature, but the methodology to best realize such capabilities is not fully developed [6]. A recurring theme from a NSF and NASA joint workshop on CES was the concept of system evolution and the need to control and capitalize on it [7]. Conceptually, all system designs can evolve; however, some systems designs lend themselves to evolution more readily than others. Four studies have identified design principles to characterize evolvability of system designs. Twelve design principles identified for military systems of systems include four strategic design principles which foster system evolvability and eight structural design principles which influence the system design directly [8]. Another study identified 24 guidelines to foster product evolvability that were categorized into one of five principles: modularity, parts reduction, spatial, interface decoupling, and adjustability [9]. Another study characterizes evolvability in terms of generality, adaptability, scalability, and extensibility within the context of space exploration systems [10], while yet another does so in terms of adaptability, flexibility, changeability, extensibility, and enhanced ability in a case study for a magnetic resonance imager [11].

While the design principles foster system design evolvability, some value judgment is needed to gauge the relative merit of system evolution design alternatives, either between two or more distinct system designs or for multiple evolutionary options for a single system design [12]. Several metrics have been suggested to gauge the potential for a particular system design to evolve. Metrics proposed and studied to varying degrees for evaluating CES design evolvability include the following:

- *System Excess* [13, 14]. Excess is the quantity of surplus in a system once the necessities of the system are met. For example, if an aircraft carrier's power plant produces 200 MW, and the carrier requires 180 MW to operate, then the excess is 20 MW. The units of excess are consistent with the feature or factor of the system being evaluated for excess (e.g., W, lb., ft^2, ft^3, \$).
- *Fitness Landscape* [4]. Product evolvability is measured on the basis of the system's design to accommodate alternative combinations of design choices of all components.
- *Filtered Out-Degree* [15]. If in addition to specifying design parameters (static representations of a system) designers also specify transition paths (dynamic change opportunities), a traditional trade space can become a trade space

network. Out-degree counts the number of potential transition paths available through the network for a given system design, and filtered out-degree applies filters for acceptable change by a particular decision-maker.

- *Visibility Matrix* [16]. Within a given system, the number of both direct and indirect dependencies that a component possesses is assessed to determine the component's visibility. The mean visibility of all components provides a measure of the evolvability for the system as a whole.
- *Interface Complexity Metric* [17]. The interface complexity is described by the material, energy, and information flows through the interface to adjacent components within a system, and the metric is calculated based on the magnitude of change in the interface parameters associated with a given design alternative.
- *Evolvability Figure of Merit* [10, 18]. Evolvability is measured as a linear combination of a particular system design's generality, adaptability, scalability, and extensibility. Each of the four constituent measures can be determined either qualitatively or quantitatively.
- *Ontological Approach* [19]. Evolvability is measured as the energy required to change system states, where the state reflects a parameterization of substantial system properties.

While conceptually general in scope, these metrics were typically studied in particular system domains or lack empirical evaluation. A broad span of 210 systems was reviewed qualitatively and identified system excess as a characteristic associated with system evolvability. The metric itself was quantitatively illustrated in a comparison of two types of aircraft carriers [13]. The use of this metric to help identify and evaluate evolvability alternatives for a system design was then illustrated on a heat gun, a coffee maker, a string trimmer, and a toy dart gun [14]. The fitness landscape metric was developed to leverage the system design abstraction of the design structure matrix (DSM), but is illustrated using only a representational DSM [4]. The filtered out-degree metric was determined for and its use in identifying and evaluating evolvability alternatives illustrated on a space tug [15, 20]. The visibility matrix was illustrated in an evolvability comparison of six releases of a commercial software product over a 15-year time span [16]. The interface complexity metric was developed and tested in two industrial case studies – a medical injector and an industrial process sensor [17]. The evolvability figure of merit is illustrated qualitatively rather than empirically in a comparison of evolvability for alternative space exploration architecture [18]. The ontological approach was illustrated using a small computer-based system [19].

Improved evolvability for a system design is viewed as desirable in general, but will typically be achieved at the price of other system design attributes. While CES that are able to evolve to meet new system requirements have more long-term value than those that are not able [21], the literature describing evolvability design principles also calls for methods to address trade-offs that arise in improving evolvability, such as the mass and volume-related overhead associated with the modularity and spatial principles [22]. Various simulation methods have been formulated and studied to determine the optimal evolvability for a system design which includes both the benefits and costs of evolvability. These methods explore a user-defined bounded trade space and evaluate viable design alternatives to determine the optimum. Three optimization approaches include:

- Epoch syncopation framework (ESF) which generates alternatives using the epoch-era construct [23];
- Excess functionality which generates alternatives from a system's hi-definition design structure matrix (HD-DSM) abstraction [24]; and
- Fitness landscape, which also generates alternatives from a system's HD-DSM, albeit uniquely from the excess functionality approach [4].

Each of these three methods is the subject of ongoing research.

Now, consider yet another path in the search for these answers – models used for evolution in nature. In nature, complex system evolution occurs seamlessly and over time ever improves performance in a dynamic environment. Using natural evolution as a model for CES design could provide insights concerning optimal evolvability. Being able to analyze system changes needed to adapt to unmet current and emerging future needs is just one job of a mathematical model of this systems generational evolution (SGE) process. Ideally, the model would also allow for analysis of any missteps in the system evolution process of previous generations to help aid in the understanding of how the system evolved and to essentially help evolve the model to be more accurate in future predictions. What is most important, though, is to come to a point where a system evolves in the right manner at the right time.

There have been applications of biological evolutionary models in business and risk management [25–27]. A goal in researching the use of biological systems to CES is to assess how a change from one generation of a system to the next influences risk or vice versa. The risk context includes both development risk and operational risk; development risk is defined to be the risk that the system will be successfully developed as planned both technically and programmatically, and operational risk is defined to be the risk that the system will prove successful in the field. In nature, development risk is addressed by controlling the rate of genetic mutation, while operational risk is addressed by the many evolutionary paths in play at any one time within a population, or survival of the fittest.

While the explicit control of technological variation would appear to be advantageous for CES relative to natural systems in the context of development risk, the ability to simultaneously evaluate many evolutionary paths appears advantageous for natural systems in the context of operational risk. One consideration is the flexibility of technology when compared to nature. In general, nature can only modify what currently exists, whereas technology can create bridges that will span over a gap to link existing technologies or to connect old technology with new [28]. These are nonnatural evolution paths. This can allow for directed changes toward predetermined evolutionary goals. It also allows various products purchased and assembled in new innovative ways, a one up on natural evolution.

52.3 Biological Models for SGE

Five biological mathematical models are considered for potential application to generational evolution for CES. These models are the Smoluchowski equation, Hardy-Weinberg equilibrium, the Price equation, the “in parallel” process, and Haldane’s model. Each of these will be briefly discussed in the following sections.

52.3.1 Smoluchowski Equation

The Smoluchowski equation is a population balance equation describing the time evolution of the number and density of particles as they coagulate and is used to model aggregation processes in polymerization and emulsification. A model was developed to investigate speciation and how co-evolution can lead to discordance based on the Smoluchowski equation. This model describes evolutionary dynamics at the phenotypic level (observable characteristics) while accounting for the underlying causal genetic mutations [29]. Equation 52.1 describes evolutionary stochastic dynamics for the case of two interacting genes in a region of DNA:

$$\frac{\partial p}{\partial t} = \frac{1}{2}\mu\nabla \cdot \left(\nabla p(\xi) - 4N_{\mathrm{e}}p(\xi)\nabla\Phi(\xi)\right) \tag{52.1}$$

where:

μ is the genetic mutation rate,
N_{e} is the effective population size,
ξ is any phenotype,
$p\ (\xi)$ is the probability of the phenotype being observed, and
$\Phi(\xi)$ is the effective potential function of the evolutionary dynamics.

This equation has potential application for modeling SGE because it helps in understanding various factors needed to maintain status quo and the process of what is changing. This model provides information on the rate of speciation in small populations. This equation also focuses on phenotypes as opposed to genotypes, which is appropriate because while we can describe the observable features of a system (e.g. airplanes have wings, automobiles have wheels, etc.), we cannot analyze the genes for a CES. The broader perspective of phenotypes would allow for a better possibility of comparison between nature and evaluation of systems, although some method to model the evolutionary dynamics for engineered system would be required.

52.3.2 Hardy-Weinberg Equation

Looking at evolution at a generational level, the Hardy-Weinberg equilibrium model can be used to find the most likely genotype frequency within a population as well as track generational changes and predict next generation probabilities [30]. Equation 52.2 describes Hardy-Weinberg equilibrium:

$$p^2 + 2pq + q^2 = 1 \tag{52.2}$$

where p and q are frequencies of the dominant and recessive genotypes, respectively. Additionally, in their research, these population geneticists understood the conditions under which evolution could and could not occur. A population will not evolve if all of the following conditions occur:

- Mutation is not occurring,
- Natural selection is not occurring,
- Population is infinitely large,
- All members of the population breed,
- All mating is totally random,
- Everyone produces the same number of offspring,
- There is no migration in or out of population.

Given the unlikely realization of all seven conditions, evolution predominates in nature. This predilection for generational evolution also predominates in the realm of engineered systems, although the generational durations vary widely relative to the intergenerational time spans for biological systems. Determination of the genome for CES will be needed for Hardy-Weinberg equilibrium to be applied to modeling SGE. However, if system components could be mapped to alleles, this simple and powerful model would allow for tracking how each successive generation of an engineered system is changing and evolving.

52.3.3 Price Equation

Another mathematical model warranting consideration is the Price equation, developed from the theory of Hamilton's rule [31]. Hamilton's rule models evolving conditions, and the Price equation describes this evolution over time depending on fitness and fidelity factors. The model, shown in Eq. 52.3, suggests that all types of selection (sociological, genetic, etc.) can be unified:

$$w_{\text{avg}} \times \Delta z_{\text{avg}} = \text{COV}(w_i, z_i) + E(w_i \times \Delta z_i) \tag{52.3}$$

In Eq. (52.3),

w is fitness and
z is the quantitative character, which can be anything such as complexity of species.

Covariance plays a key role in showing that what matters is statistical links between genotypes of one generation to the next and not a shared lineage [31]. This model unifies the various types of selection that can occur in the evolution process. If it is possible in the future to map engineered systems to a genotype, this equation would help in identifying what components are evolving regardless of the type of selection, more accurately predicting evolutionary vectors for engineered systems on the basis of technology advances in particular disciplines, helping to answer questions such as the impact of 3D printing or the Internet-of-things on future generations of existing engineered systems.

52.3.4 "In-Parallel" Process Equation

As previously mentioned, natural system evolution possesses an advantage relative to engineered system evolution in that nature has the luxury of experimenting with many evolutionary changes in parallel. While nature lacks the leapfrog capacity of technological innovation in CES, she makes up for that with the sheer number of mutations and the patience to let natural selection sort out the best designs over time. An "in-parallel" evolutionary model is given in Eq. (52.4) and is built upon the idea that an advantageous mutation at for one gene becomes accepted within a population concurrent with the evolutionary process underway for other gene [32], with the effect of dramatically shortening the timescale or number of generations required for superior phenotypes to emerge:

$$G \cong \frac{\log L + \gamma}{\log\left(\frac{K}{K-1}\right)} + \frac{1}{2} \tag{52.4}$$

where

G is the number of generations,

L is the length of the genomic "word" or number of individual genes,

K is the number of possible "letters" that can occupy any position in the word or number of possible expressions of an individual gene.

This model has the advantage of mimicking the system development process in which most complex system designs are a mix of heritage elements and new elements. In complex system design, and in successive generations of evolution for complex system design, it is indeed a rarity to realize only one change into the system design; rather, multiple changes are incorporated in parallel. This model holds potential to optimize the number of generations needed to field a forecast portfolio of evolutionary upgrades.

52.3.5 Haldane's Equation

J.B.S. Haldane developed mathematical models of evolution in natural systems that focused on phenotypes and evolution across generations rather than across time. His model, given in Eq. 52.5, quantifies phenotypic variance for a species at two instances in time, and then normalizes the timescale in terms of the number of generations. A unit of measure using this approach, later dubbed the Haldane, is defined as change by a factor of one standard deviation per generation [33]. Haldanes allow for system evolution comparisons on a generation level instead of time since it is possible for different systems to evolve at different speeds:

$$h = \left(\frac{\left(\frac{\ln x_2}{s_{\ln x}}\right) - \left(\frac{\ln x_1}{s_{\ln x}}\right)}{t_2 - t_1} \right) \tag{52.5}$$

where:

h is the rate of generational evolution,

ln x_1 and ln x_2 are representative measurements to quantify the variation in phenotypes at times t_1 and t_2, respectively, and

$s_{\ln x}$ is the pooled standard deviation of the measurements.

The intrinsic rate is "the average difference between successive generations" [33]. This rate would give a forecast of how the next generation would progress in the evolutionary process. Assessments indicate that a simple answer to how fast evolution occurs in the natural world is approximately an intrinsic rate of "one-tenth (10^{-1})" of a standard deviation per generation, which places an upper limit on the rate of evolutionary adaptation for a population.

This model has the advantage of being based on phenotypes only, which can be measured for CES, and hence not requiring genotypic specification of a CES, for which no method currently is known. Furthermore, it deals with generational timescales, fostering application to different families of engineered systems for which the generations vary. For instance, the generational timescale for the Lockheed Martin C-130 airplane is considerably longer than that for the Apple iPhone. As such, application of this model is more straightforward than the previous models listed. If a single intrinsic rate or a family of intrinsic rates could be determined, this model could describe the risk frontier for various alternative system designs in an analysis of alternatives. Knowing how fast and how much a system can reasonably evolve would allow SGE to occur in the most efficient and effective way possible.

52.4 Phenotypes and Genotypes in a Complex Engineered Systems Context

The foregoing section described five models of evolution in natural or biological systems, and the terms phenotype and genotype were used often. Indeed, these terms have their origins in the characterization of nature, so that is to be expected. Phenotypes represent observable characteristics such as wings, hair color, an opposable thumb, and so forth. Genotypes represent the genetic aspect. In nature, evolution occurs as a result of genetic mutation – nature varies genotypes, and the resulting phenotypes influence natural selection. The superior phenotypes in turn pass on the associated genotypes to successive generations. To productively utilize any of the five models presented previously for modeling generational evolution of CES, some meaningful interpretation of phenotypes and genotypes to CES will be necessary.

52.4.1 A Classification System of Systems

Over the past few centuries, natural scientists have organized the variety of plant and animal life on the basis of phenotypes as depicted in Fig. 52.1. This biological taxonomy in essence groups biological organisms on the basis of shared characteristics and gives names to those groups. These groups can in turn be aggregated to form super groups or refined to form subgroups, resulting in a taxonomic hierarchy of Fig. 52.1.

Might such a taxonomy of CES be possible? If so, a classification of the phenotypes that characterize the myriad forms of CES would result, providing a basis for quantitative assessment of phenotypes for any particular CES concept. If a usable classification system can be developed, it will be possible to examine new system concepts and to determine where on the CES taxonomy spectrum it is most

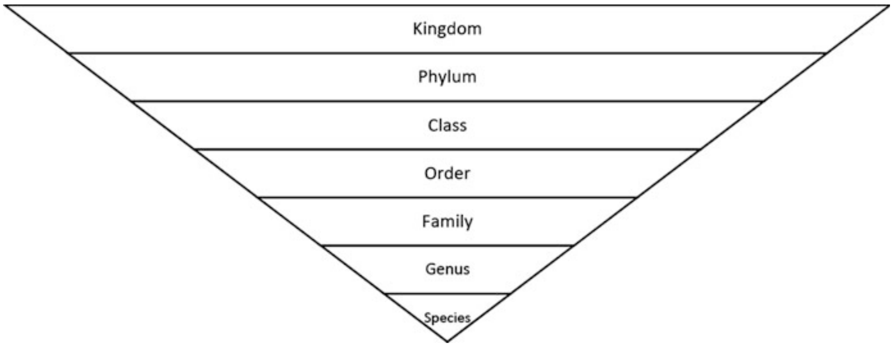


Fig. 52.1 Biological taxonomy

likely to fall based on similarities to other existing CESs. The overlaps of technology used between potential existing systems, the technological differences, as well as the planned evolutionary goal may give insight into how to evolve to a prototype CES. Hence, in addition to feeding the natural system evolutionary models previously discussed, such a CES taxonomy would inform the planning process for new CES developments. To illustrate, consider a new system topology such as the V-22 tilt-rotor aircraft. In planning the V-22 system development, where do the project manager and systems engineers begin for this new configuration? Phenotypic assessment suggests shared characteristics with a helicopter and a fix wing airplane, and by measuring the difference between these two known topologies (fixed wing airplane and helicopter), the evolutionary models can estimate the scale of the evolutionary advance represented by this new CES, and inform plans regarding uncertainties and risk in the development timescale and approach. With this knowledge, it may be possible to predict the evolution of the new system. This would allow one to pinpoint where things need improvements and/or where potential path direction changes will occur. Also, by understanding phenotypes, otherwise unforeseen applications for new technologies will become evident, enabling technology infusion in systems that are not necessarily closely related by business unit.

52.4.2 The CES Genome

Whereas biological phenotypes readily lend themselves to interpretation in CES, a straightforward interpretation of biological genotypes is more elusive. Whereas nature experiments with genetic mutation and lets natural selection evaluate the resultant phenotypes, CES have no genotypes. Or do they? If the genotypes are interpreted as the information encoding for a biological system to be realized, then the algorithms and engineering drawings may serve as proxies for the CES genotypes. Unfortunately, the algorithms and engineering drawings are developed as part of the CES design and development process, rather than preceding it. Equally unfortunately, the algorithms and drawings lack the coherence of a string of DNA for a biological system. However, the relatively recent emergence of integrated systems models utilizing tools based on languages such as SysML and AADL offers the potential to at least partially address both shortcomings. These systems modeling languages provide the capability to model a system at varying levels of abstraction, allowing an integrated system model to be coherently developed at a high degree of abstraction and then progressively refined in detail within that architectural construct. An integrated systems model, at any given level of abstraction, which holds the information needed to simulate the system in operation is then arguably serving as the CES genome. Successive refinement will add detail and improve the precision and accuracy of the integrated system model, but should not fundamentally alter the simulated behavior.

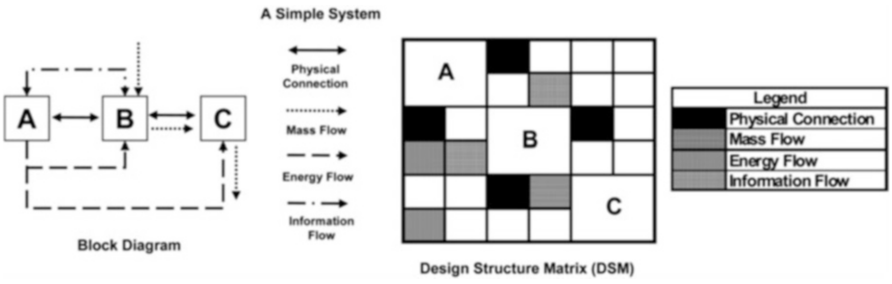


Fig. 52.2 Block diagram and DSM of a simple system [35]

If an integrated system model can serve as a proxy for the CES genome, the challenge then becomes one of transforming the information comprising the integrated systems model into genotype equivalents. One potential strategy is to represent the integrated systems model structural elements and their relationships as a high-definition design structure matrix (HDDSM). The HDDSM is an evolved product representation model that captures a spectrum of interactions between components of a product, such that the characteristics of product architecture can be assessed and compared [34]. In such a representation, entries of a 1 or 0 in the matrix mark the presence or absence of a particular interaction between two components of the system – intuitively analogous to genetic information. Clearly, the HDDSM is not sufficient as a CES genome, for the system behavioral aspects within the integrated system model are not well characterized. The HDDSM for a CES does not include the behavioral information needed for a system simulation. Still, it may provide a credible first step and offer enlightenment concerning how to transform the integrated systems model into genotype equivalents (Fig. 52.2).

52.5 Remarks

CES design and development is rife with challenges for the systems engineers, from missing requirements to integration challenges to inherent uncertainties in maturing new or updated technologies. Further complicating the development process is the dynamic nature of the systems development environment – the operational need will evolve with time, sometimes drastically, and new technologies will become available. Nature has many things yet to teach in regards to finding balance how and when to incorporate changes to successfully field complex systems. Biological mathematical models provide good stepping-stones along the path of understanding SGE. Models of evolution provide insight into how a CES could and/or should change. These same models could be used to analyze previous systems for what went right and wrong. Being able to compare existing systems within a classification system, one could hypothesize how evolution would occur in new systems or how the existing systems could evolve into a new system.

In researching how to mathematically model biological evolution, existing models seem to be plentiful and good at mimicking portions of the evolutionary process. All models described in this chapter have potential to help aid in modeling the evolution of CES, but there is not a correct one per se. More research is needed to determine which models are most relevant to CES generational evolution. Clearly, a CES equivalent to the genotype in biological systems will be needed to productively employ most of the existing models, and the pathway to such an analog is not clear. One could argue that a taxonomy of biological evolution models would be of great benefit when it comes to creating the model for systems design. Given a CES taxonomy, a classification of systems would create a bridge between biological and technological systems. With the phenotype and genotype bridges between biological systems and CES, correlations would allow models that already harness elements of evolution from nature to be applied to CES, leading to more "natural" evolution of CESs.

References

1. Thompson L (2016) New pentagon contract signals lockheed Martin's C-130 airlifter is headed for 100 years of service. Forbes
2. Blake A (2015) Black hawk drone: unmanned chopper passes critical Pentagon test. p. - Washington Times
3. Tackett MWP (2013) A mathematical model for quantifying system evolvability using excess and modularity
4. Luo J (2015) A simulation-based method to evaluate the impact of product architecture on product evolvability. Res Eng Des 26(4):355–371
5. Madni AM (2012) Adaptable platform-based engineering: key enablers and outlook for the future. Syst Eng 15(1):95–107
6. Simpson TW, Martins JRRA (2011) Multidisciplinary design optimization for complex engineered systems: report from a National Science Foundation workshop. J Mech Des 133 (10):101002–101002
7. NSF/NASA Workshop on the Design of Large-Scale Complex Engineered Systems - From Research to Product Realization in 12th AIAA Aviation Technology, Integration, and Operations (ATIO) Conference and 14th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, American Institute of Aeronautics and Astronautics, (2012)
8. Ricci N, Rhodes DH, Ross AM (2014) Evolvability-related options in military systems of systems. Procedia Comput. Sci. 28:314–321
9. Keese DA, Tilstra AH, Seepersad CC, Wood KL (2007) Empirically-derived principles for designing products with flexibility for future evolution, presented at the ASME 2007 international design engineering technical conferences and computers and information in engineering conference. Las Vegas, Nevada, pp 483–498
10. Christian JA III (2004) A quantitative approach to assessing system evolvability. NASA Johnson Space Center, Houston
11. Borches PD, Bonnema GM (2008) On the origin of evolvable systems: evolvability or extinction, presented at the seventh international symposium on tools and methods for concurrent engineering. TMCE, Izmir
12. Tilstra AH, Seepersad CC, Wood KL (2009) Analysis of product flexibility for future evolution based on design guidelines and a high-definition design structure matrix. In: ASME 2009

international design engineering technical conferences and computers and information in engineering conference, 951–964
13. Tackett MWP, Mattson CA, Ferguson SM (2014) A model for quantifying system evolvability based on excess and capacity. J Mech Des 136(5):051002–051002
14. Cansler EZ, Ferguson SM, Mattson CA Exploring the relationship between excess and system evolutions using a stress-test, ASME proceedings 27th international conferences design theory methodology. p. V007T06A041, Aug 2015.
15. Ross AM, Rhodes DH, Hastings DE (2008) Defining changeability: reconciling flexibility, adaptability, scalability, modifiability, and robustness for maintaining system lifecycle value. Syst Eng 11(3):246–262
16. MacCormack A, Rusnak J, Baldwin CY (2007) The impact of component modularity on design evolution: Evidence from the software industry, Harvard business school technology and operations mgt unit research paper no. 08–038
17. Hölttä-Otto K (2005) Modular product platform design. Helsinki University of Technology, Espoo
18. Christian JA III, Olds JR (2005) A quantitative methodology for identifying evolvable space systems. In: 1st space exploration conference: continuing the voyage of discovery, Orlando, Florida, p. 2543.
19. Rowe D, Leaney J (1997) Evaluating evolvability of computer based systems architectures-an ontological approach. In: International conference and workshop on engineering of computer-based systems 1997 PRO, p. 360–367
20. Fulcoly DO (2012) A normative approach to designing for evolvability : methods and metrics for considering evolvability in systems engineering, Thesis, Massachusetts Institute of Technology
21. Bloebaum CL, McGowan AR (2012) The design of large-scale complex engineered systems: present challenges and future promise. In: Proceedings of the 14th AIAA/ISSMO multidisciplinary analysis and optimization conference, Indianapolis, in, paper no. aiaa-2012–5571
22. Tilstra AH, Backlund PB, Seepersad CC, Wood KL (2015) Principles for designing products with flexibility for future evolution. Int J Mass Cust 5(1):22–54
23. Fulcoly DO, Ross AM, Rhodes DH (2012) Evaluating system change options and timing using the epoch syncopation framework. Procedia Comput Sci 8:22–30
24. Watson JD, Allen JD, Mattson CA, Ferguson SM (2016) Optimization of excess system capability for increased evolvability. Struct Multidiscip Optim 53(6):1277–1294
25. Bonabeau E (2007) Understanding and managing complexity risk. MIT Sloan Manag Rev 48 (4):62
26. Rammel C, van den Bergh JCJM (2003) Evolutionary policies for sustainable development: adaptive flexibility and risk minimising. Ecol Econ 47(2–3):121–133
27. Lo AW The adaptive markets hypothesis: market efficiency from an evolutionary perspective. In: Social science research network, Rochester, NY, SSRN scholarly paper ID 602222, Oct. 2004.
28. Burris K, Thomas LD (2016) The palm, a systems generational evolution story.pdf. In: Presented at the conference on systems engineering research, Huntsville
29. Khatri BS, Goldstein RA (2015) A coarse-grained biophysical model of sequence evolution and the population size dependence of the speciation rate. J Theor Biol 378:56–64
30. Relethford JH (2012) Hardy–weinberg equilibrium. In: Human population genetics, John Wiley & Sons, Inc. p. 23–48.
31. Whittaker D (2012) BEACON researchers at work: mathematical modeling of evolution | BEACON
32. Wilf HS, Ewens WJ (2010) There's plenty of time for evolution. Proc Natl Acad Sci 107 (52):22454–22456
33. Gingerich PD (1993) Quantification and comparison of evolutionary rates. Am J Sci 293:453–478

34. Tilstra AH, Seepersad CC, Wood KL (2012) A high-definition design structure matrix (HDDSM) for the quantitative assessment of product architecture. J Eng Des 23 (10–11):767–789
35. Suh ES, Furst MR, Mihalyov KJ, de Weck OL(2008) Technology infusion: an assessment framework and case study. In: ASME Proceedings of the 20th international conferences design theory methodology. DTM, p. 297–307

Chapter 53
Evaluating How System Health Assessment Can Trigger Anticipatory Action for Resilience

David Lowe, Philip Oliver, Gerald Midgley, and Mike Yearworth

Abstract In 2014, the Defence Science and Technology Laboratory developed and implemented a novel approach to assess the system by which the United Kingdom Ministry of Defence delivers infrastructure projects and services. This approach brought together existing methods to constitute a hybrid problem structuring method that offered the potential to trigger anticipatory intervention by focusing on the health as opposed to the performance of this system. This paper revisits the initial assessment to examine whether use of the method has led to increased system resilience, and in particular to understand what it was about the method that helped to deliver benefits. Insights with regard to the structures and processes necessary to enable anticipatory action for resilience are presented.

Keywords Systems thinking • Resilience • System health monitoring • Anticipatory systems

D. Lowe (✉)
Defence Science and Technology Laboratory, Salisbury, UK

University of Bristol, Tyndall Avenue, Bristol BS8 1TH, UK

Centre for Systems Studies, Business School, University of Hull, Cottingham Road, Hull HU6 7RX, UK
e-mail: dalowe@dstl.gov.uk

P. Oliver
Ministry of Defence, Horse Guards Avenue, London SW1A 2HB, UK

G. Midgley
Centre for Systems Studies, Business School, University of Hull, Cottingham Road, Hull HU6 7RX, UK

School of Innovation, Design and Engineering, Mälardalen University, Eskilstuna, Sweden

Victoria Business School, Victoria University of Wellington, Wellington, New Zealand

School of Political and Social Sciences, University of Canterbury, Christchurch, New Zealand

School of Agriculture and Food Sciences, University of Queensland, Brisbane, QLD, Australia

M. Yearworth
University of Exeter, Rennes Drive, Exeter EX4 4PU, UK

A.M. Madni et al. (eds.), *Disciplinary Convergence in Systems Engineering Research*, https://doi.org/10.1007/978-3-319-62217-0_53

53.1 Introduction

This paper follows up on a system health assessment conducted within the United Kingdom (UK) Ministry of Defence (MOD) in 2014 to guide anticipatory intervention within its Defence Infrastructure System.

The Defence Infrastructure System boundary encompasses the Defence Infrastructure Organisation (DIO) together with the elements of MOD Head Office that provide governance for the operation of the DIO ("Governor" role) and the Armed Forces and other elements of MOD that provide infrastructure requirements for the DIO to deliver against ("Customer" role). The DIO was created in 2011 to *"coherently manage delivery against MOD's infrastructure requirements"* [1] and is responsible for building, maintaining, and servicing the infrastructure necessary to enable MOD personnel – both military and civilian – to live, work, train and deploy both in the UK and overseas.

To drive further efficiency and effectiveness, in 2014, MOD decided to introduce a strategic business partner within the DIO to act as the senior management team and inject knowledge and expertise developed in the private sector. As part of the preparations for this introduction, the Defence Science and Technology Laboratory (Dstl) was tasked to assess the health of the Defence Infrastructure System to identify major strengths, weaknesses and uncertainties, and also to make recommendations to be implemented as part of a broad reform programme [2].

This study identified eight key actions that should be taken with the purpose of improving the health (and consequently long-term performance and resilience) of the Defence Infrastructure System that were taken forward in the Defence Infrastructure System Programme. Immediate feedback was very positive, but in November 2016, almost exactly 2 years after the results were briefed, the UK National Audit Office published a report [3] which found that whilst MOD *"has started to improve its management of the defence estate . . . the department has not yet set out how it will fully address the significant challenges it faces sustaining the whole of its estate and the resulting risks to military capability"* and went on to make a number of recommendations for further intervention. This suggests that the original health assessment failed to trigger all the changes necessary – because not all appropriate actions were identified by the initial assessment and/or because their implementation is not yet complete.

It is well established that most change programmes fail, with failure rates as high as 70% (Kotter [4]). This chapter examines the reasons why the identified actions seemingly have failed to achieve the desired outcomes in terms of two dimensions – desirability and feasibility – following Peter Checkland's criteria for identifying purposeful actions as part of his Soft Systems Methodology [5]. In terms of desirability, this chapter considers where the actions have had a positive effect (i.e. as designed); where the actions have had a negative effect (i.e. where unintended consequences have come into play – as per the 'fixes that fail' systems archetype [6]); where they have had no effect at all (i.e. the action was not necessary); and where additional actions should have been recommended (i.e. the action set was not sufficient). In terms of feasibility, this chapter considers which actions proved straightforward to implement and which actions proved difficult (such that implementation was significantly compromised or failed altogether) to better understand barriers and enablers to progress.

Taken together, the results will help provide answers to the research question posed recently as part of a movement to advance anticipation as a multidisciplinary research field: What structures and processes are necessary for anticipatory action? [7]. The chapter now proceeds to:

- detail why anticipatory action is important for a resilient system and how internal health assessments can be used to trigger such action (see Sect. 53.2);
- revisit the assessment that was conducted in 2014 and the recommendations that were made (see Sect. 53.3);
- evaluate the desirability and feasibility of the outcomes that followed from these recommendations in the light of events that have followed via stakeholder interviews (see Sect. 53.4 Evaluation);
- discuss the insights generated by the evaluation and identify areas for further work (see Sect. 53.5); and
- summarize the findings and highlight the key points (see Sect. 53.6).

53.2 Literature Review

The term 'resilience' stems from Latin (resiliens) and originated from the physical sciences where it is commonly used to refer to the '*ability of a substance or object to return to its original shape after being bent, stretched, or pressed'* [8]. Resilience has been the subject of much scholarly debate in the systems literature since Holling [9] first introduced the concept in a systems context in 1973 where he defined resilience of an ecosystem as 'the measure of the ability of these systems to absorb changes of state variables, driving variables, and parameters and still persist'. In particular, a competing 'engineering resilience' paradigm has emerged that is distinct from Holling's 'ecological resilience'. Where ecological resilience (also referred to as static resilience or robust resistance) involves maintaining the existence of function and is typically measured in terms of the system's ability to resist disturbances, engineering resilience (also referred to as agile adaptability or dynamic resilience) involves maintaining efficiency of function, including by taking action to achieve a more desirable state in advance of a disturbance, and is typically measured in terms of speed of return to a state of efficiency [10].

In the public sector, where threats to existence are rare (whilst a government department may be sanctioned for a lack of efficiency it almost always persists, and especially a Department of State such as MOD), it is the concept of 'engineering resilience' that is the more relevant. Moreover, it is vitally important that government departments such as the UK MOD continue to deliver essential services regardless of changes in operating conditions and that they take pre-emptive action to ensure this. In particular, it is important for the defence and security of the UK that MOD is able to deliver the outputs specified in a set of military tasks: defending the UK and its overseas territories; providing strategic intelligence; providing nuclear deterrence; supporting civil emergency organizations in times of crisis; defending our interests by projecting power strategically and through expeditionary interventions; providing a defence contribution to UK influence; and providing security for stabilization [11].

To maintain efficiency of function under the engineering resilience paradigm, it has been suggested that a system should have the ability to anticipate, monitor, respond and learn [12]. The collation and processing of information is critically important for each of these cornerstones, and the impact of introducing information systems on an organization's resilience has been identified as a fruitful direction for research in this area [13]. Such information systems should enable the collation and processing of contextual information that is both external and internal to the system if they are to enable pre-emptive anticipatory action [14, 15] that will ensure resilience in the engineering resilience sense. Whilst MOD has a number of structures and processes dedicated to the collation and processing of external information, there is less dedication to the collation and processing of internal information. It is postulated here that system health assessments can provide a useful complement to existing performance, risk and audit reporting and make for a more complete set of internal contextual information from which MOD can take anticipatory action.

In the field of organizational management, health is being increasingly recognized as a lead indicator for performance – 'Performance is about delivering (financial) results in the here and now. Health is about the ability to do it year in, year out' [16] – that has the potential to transform management approaches from reactionary to anticipatory. Whilst a number of approaches have been developed to enable health assessment in public, private and third sectors – see previous CSER paper for a description of these [17] – there appears to have been no attempt to evaluate their effectiveness in a longitudinal sense.

This chapter revisits the initial assessment conducted in 2014 to understand why the action that was recommended has not fully achieved the benefits that were anticipated. Of particular interest is what it was about the method that was developed and implemented that could be improved. For example, did the apparent failure stem from the way in which the process was managed (i.e. building commitment of individuals, establishing multiorganizational teams) or how the content was managed (i.e. summary visualization enabled ready appreciation of key issues, hierarchical representation enabled linking of point issues in wider context) or both [18]? In this way, insights are offered against the research question highlighted earlier: What structures and processes are necessary for anticipatory action? [7].

53.3 Initial Assessment

The initial assessment in 2014 involved the development and implementation of a hybrid problem structuring method (PSM) that combined hierarchical process model (HPM) with viable system model (VSM) methods in a pluralist [19] (or multimethodological [20]) design.

HPM was developed at the University of Bristol based on a strong process-based view of system description [21]. The HPM tree structure provides a conceptual schema for establishing the processes required to achieve a transformation. Leaf processes are described using gerund form to stimulate a degree of creativity in the modelling (gerund forms have no subject, and the performer is not specified) and are scored using an Italian Flag scale that details what is known to be good about a process (green), what is known to be bad about a process (red) and what is uncertain or unknown (white). These scores are then typically aggregated using sufficiency and necessity conditions drawing upon a variation of the interval probability theory implemented in what is termed the Juniper algorithm [22].

Eliciting the hierarchical system structure typically proceeds from a top-level transformational process through repeated questioning of 'how', until there is no longer a process answer to these 'how' questions. However, in this application, the HPM structure was developed through the application of VSM at two different levels to examine the Defence Infrastructure System in the wider context of the Defence Enterprise (see Fig. 53.1). The VSM was developed by Beer [23–25] by applying cybernetic theory (regarding the flow and use of information for regulation and control) to the management of organizations. He established that the

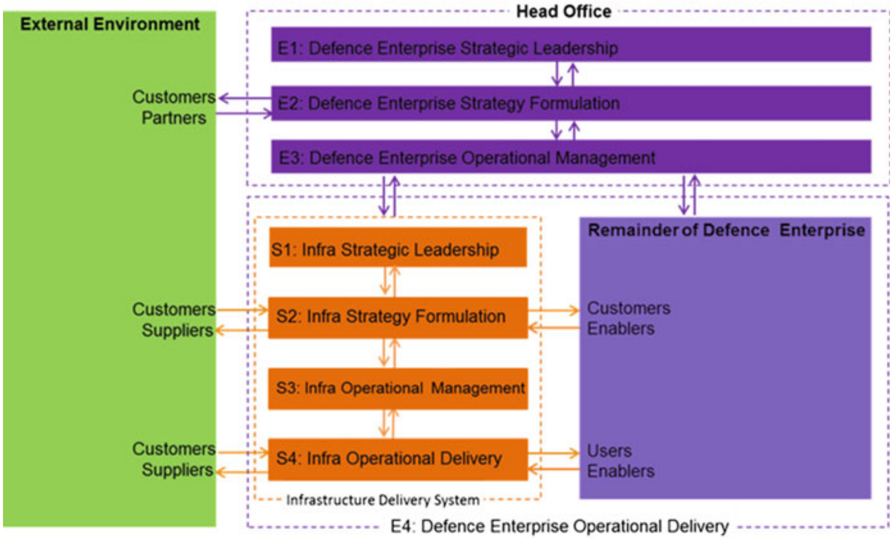

External Environment
Head Office
E1: Defence Enterprise Strategic Leadership
Customers
Partners
E2: Defence Enterprise Strategy Formulation
E3: Defence Enterprise Operational Management
S1: Infra Strategic Leadership
Remainder of Defence Enterprise
Customers
Suppliers
S2: Infra Strategy Formulation
Customers
Enablers
S3: Infra Operational Management
Customers
Suppliers
S4: Infra Operational Delivery
Users
Enablers
Infrastructure Delivery System
E4: Defence Enterprise Operational Delivery

Fig. 53.1 Assessing the infrastructure delivery system in the context of the defence enterprise.

viability of an organization in its environment is dependent on the capacity of, and strong links between, five key system elements – identity (S5), intelligence (S4), control (S3 including audit S3*), coordination (S2) and operations (S1). These 'invariances' provide a functional framework for the design of a new system and/or the assessment of an existing system, and can be applied recursively to model organizational scale and complexity.

However, the application of VSM in its standard form was found to be problematic. Stakeholders had difficulty in discriminating between Beer's S3 (control), S3* (audit) and S2 (coordination), because they regarded these functions as largely indistinguishable within the role of management. Consequently, these functional requirements were collapsed into a single layer to yield an adapted VSM focused on four key functions (with associated subfunctions):

- Strategic leadership (Beer's S5): Setting strategic direction; setting strategic incentives; and managing strategic performance and risk.
- Strategy formulation (Beer's S4): Capturing inputs and constraints; setting strategy; and monitoring strategy implementation.
- Operational management (Beer's S3, S3* and S2): Setting management direction; managing performance and risk; assuring delivery coherence; and assuring and auditing performance.
- Operational delivery (Beer's S1): Understanding user requirements; understanding supplier capabilities; delivering projects and services; and managing delivery performance and risk.

The assessments were undertaken via two workshops: one focusing on 'Defence Enterprise Operational Management' (function E3) related to infrastructure delivery and the other focusing on 'Infrastructure Delivery System' (functions S1–S4). The workshop participants were those stakeholders identified as having responsibility for and/or experience of, delivering key elements within functions E3 and S1–S4, as well as those with experience of setting requirements and receiving services as a customer (principally the Armed Forces). Eight actions were identified based upon five key weaknesses and three key areas of uncertainty (see Table 53.1). These recommended actions were readily accepted by the senior customer (who was not

Table 53.1 Recommended action for intervention within Defence Infrastructure System.

#	Recommended Actions	Systemic Problem
1	Define scope of infrastructure delivery system	Uncertainty
2	Test, formalise and communicate internal interfaces	Uncertainty
3	Test, formalise and communicate external interfaces	Uncertainty
4	Accelerate roll-out of management information systems	Weakness
5	Link infrastructure delivery to defence objectives	Weakness
6	Establish incentives for 'Defence first' behaviours	Weakness
7	Manage infrastructure delivery system as a whole	Weakness
8	Develop intelligent customer status (Head Officeand Armed Forces)	Weakness

party to any of the workshop discussions) and used to define the forward work plan for the Defence Infrastructure System Programme.

The immediate cross-sectional evaluation conducted using the framework developed by Gerald Midgley and his co-authors [26] revealed very positive stakeholder views, with two main benefits being identified from the perspective of follow-on work to establish a maturity model and track progress of actions being taken. First, the work had brought the stakeholders together to understand the system: '*The Dstl work made a real difference in the success of the Defence Infrastructure System Programme definition stage. In particular, it enabled us to arrive at an agreed maturity model much more quickly than would otherwise have been the case as many of the key stakeholders had already been through the thought process that led them to understand the functions within the system'.* Second, the work identified key areas for intervention: '*In addition, your work with them around the Italian Flag assessment resulted in a common understanding of system weaknesses. This enabled us to reach a rapid and robust consensus as to the current state of the Infrastructure System and priorities for corrective action'.*

53.4 Evaluation

The research question posed here – What structures and processes are necessary for anticipatory action? – has necessitated the design and implementation of a longitudinal evaluation method. This method draws upon the principles of realistic evaluation [27] in that it considers an outcome (O) to be the result of the application of a given method (M) in a given context (C), or C + M = O for short. This construct provided the basis for six structured interviews with stakeholders drawn from all aspects of the Defence Infrastructure System – two from Head Office, two from Armed Forces and two from DIO – where the interview team asked a set of open questions relating first to context, then to outcome and finally to method. Of these six interviews, four were conducted with staff who were involved in the initial assessment and two were conducted with staff who were new in their posts (both in the Armed Forces).

In response to the opening set of questions, stakeholders reported that the context over the intervening 2 years had been challenging in three key aspects, and these had limited the implementation of the actions identified in 2014. First, it was noted that the DIO had experienced service provision issues at the Infrastructure Operational Delivery level where customers were dissatisfied with the service delivered through a number of related contracts and that this had led to relationships with the Armed Forces to become strained. Second, and as a direct result of these service provision issues, it was highlighted that Head Office had been unable to maintain a broad system-wide perspective at the Infrastructure Strategic Leadership level and instead had typically focused in one or two areas. The fact that this occurred despite MOD establishing a dedicated change programme highlights the acute nature of these service provision issues. Third, it was observed that the installation of a strategic business partner at the Infrastructure Operational Management level had yet to realize all of the expected benefits and had in some areas led to significant additional complications. More positively though, the stakeholders reported that, compared to the assessment made for September 2014, there were now reduced levels of uncertainty owing to the work conducted by the Defence Infrastructure System Programme.

The way in which this work had/had not delivered outcomes was the subject of the next section of the interviews which examined how the actions detailed in Table 53.1 had been progressed (recall that each was designed to be both feasible and desirable). It was established that seven out of these eight actions – #1, #3, #4, #5, #6, #7 and #8 – had been progressed substantially and that each had delivered a positive outcome (i.e. they were indeed both feasible and desirable). For example, it was evidenced that improvements in the collation, processing and dissemination of management information have directly led to improvements in decision-making with specific reference to estate life cycle, maintenance backlogs and footprint rationalization. Whilst feasible, Action #2 had proven to be less desirable than the other recommended actions and so had been deprioritized within the Defence Infrastructure System Programme. It was also established that the recommended actions were a comprehensive set and this was attributed to the broad nature of the actions that meant that even with the benefit of hindsight nothing additional could and should have been recommended. It was acknowledged however that not all of the actions identified as both desirable and feasible had yet to fully pay off – primarily due to the challenging context described above and also due to their long-term nature.

The final section of each interview examined what it was about the method that had delivered a comprehensive set of actions that had proven to be largely both desirable and feasible. The interviewees identified three key aspects. First, the interviewees found that by engaging and gathering together a broad range of stakeholders to exchange views, and from their unique perspectives on the current ‘as is’ operating model, the method yielded high levels of stakeholder engagement that persisted over time. Second, the interviewees found comparing the intended ‘to be’ operating model against a pre-established framework in a structured way to be helpful in identifying important areas of uncertainty where more definition was required. Third, the interviewees found conducting the baseline assessment that identified both strengths and weaknesses (in addition to these uncertainties) to be helpful in establishing a common stepping off point from which direction could be set with confidence.

53.5 Discussion

Whilst cross-sectional evaluation is useful, in that it provides benefit to researchers by accessing different perspectives other than their own, longitudinal evaluation goes further to provide complementary benefits (as acknowledged by Midgley and his co-workers) [26]. Longitudinal evaluations have benefits both for researchers and for participants. For researchers, such evaluations can reveal the blockers and enablers for long-term impact that follow from their initial work. For participants, such evaluations provide a useful stimulus to revisit original issues, reflect and perhaps reinvigorate or at least maintain momentum. Both types of benefits were realized in this case.

This longitudinal evaluation confirms the key finding from the cross-sectional evaluation of just how suited problem structuring methods are to guiding intervention in messy, problematical situations involving multiorganizational groups [28–30]. This stems from enabling a social process where multiple stakeholders are engaged, and they are encouraged to contribute from their perspectives whilst simultaneously providing a structure for handling content that supports effective discussion and/or dialogue. The benefits that stem from the simultaneous management of process and content are well known [18] (sometimes with the injection of substantive expertise [31]). In this instance, stakeholders particularly valued the structure provided by the VSM, as it provided a useful handrail for guiding discussion and assessment.

Both this longitudinal evaluation and the initial cross-sectional evaluation highlighted the potential for Italian Flag assessments to act as powerful boundary objects where the model provides a number of affordances [32] that enable multiorganizational groups to constructively focus on the assessment and what actions to take rather than argue from entrenched positions – it was interesting to record one interviewee characterizing this in their own terms as '*corralling whinging for use as a basis for action*'. In particular, the stakeholders appreciated the way in which this assessment scheme provides for the visualization of uncertainty – the white one of the Italian Flag – in addition to the balance of strengths (green) and weaknesses (red). In some instances, this represented situations where insufficient information was available, and in others it represented situations where participants agreed to disagree, but in all instances, there was a commitment to work to reduce the uncertainty where it was assessed to have significant impact on decision-making. The identification and treatment of uncertainty is recognized as a key aspect of strategic decision-making [33], and it was interesting to note that work subsequent to this assessment used a maturity model approach to track progress against benchmark levels set on a 5-point Likert scale against a number of functional requirements – see [34] for a review of maturity model approaches. Whilst such approaches are well suited to setting targets and tracking progress, they do not provide for recording and treatment of the uncertainty.

This focus on health as opposed to performance was found to be helpful in that it somewhat distanced participants from the symptoms in the here and now to focus on the underlying causes. In particular, it enabled the senior leaders involved to 'step back' and to reflect on the effect that actions were having, and so to identify the need for alternative and/or additional intervention that otherwise would not have been brought about. The areas identified through the assessment have been welcomed as evidence-based interventions that, when fully implemented, will increase resilience by reducing the likelihood of risks materializing as issues.

Such a systems approach has since been applied elsewhere in MOD to good effect, most notably for acquisition and also for Permanent Joint Head Quarters and Head Office organizations. As the UK government continues its move towards commissioning the delivery of public services, it appears that opportunities for adding value via this type of assessment will continue to expand. Further, the flexible nature of this approach to assessment means that it lends itself to examination of a broad range of interface types (contractual, inter-organization or intra-organization) and so could provide the basis for a wide spectrum of use in organizational settings.

It should be noted that the development and implementation of this novel approach to health assessment took place during a period of extreme change in Defence Infrastructure System. Whilst initial exploration of the broader adoption of this approach in other organizational settings (and in particular public sector organizations) has shown great potential, to realize maximum benefit, the organization in question needs to be stable enough for the senior managers to take on responsibility for delivering broad-based systemic change as opposed to being limited to addressing pressing issues that are narrow in scope. The inability of senior managers to act systemically proved to be a major blocker in this particular case.

53.6 Conclusions

This chapter has revisited the initial assessment carried out for the United Kingdom Ministry of Defence in 2014 with regard to the way in which it delivers infrastructure projects and services to evaluate whether it has led to increased system resilience through the triggering of anticipatory action. The analysis of a number of interviews conducted with stakeholders suggests that whilst seven of the eight actions identified in 2014 have brought positive outcomes, the extremely difficult operating context – with the MOD having to deal with a number of pressing performance issues as detailed in the NAO report [3] – has limited MOD's ability to drive systemic change and fully realise the associated health benefits.

This chapter has identified a number of insights with regard to the structures and processes necessary to enable anticipatory action for resilience that appear to be readily transferrable to other settings. These comprise the needs to: (i) step back to consider system health (and not system performance) to identify actions necessary to address the underlying systemic issues; (ii) engage and bring stakeholders together to exchange views and jointly identify necessary actions to engender trust and commitment; (iii) provide a means for enabling stakeholders to critique both the 'as is' and 'to be', and to summarise, this is a meaningful way that can be readily accessed by others to help communicate what needs to be done; (iv) recognize uncertainty where it exists to ensure that action is taken to reduce it (in addition to action to address weakness); and (v) revisit assessment over time to maintain momentum in implementation.

Acknowledgements This chapter has substantially benefited and been shaped by collaboration with Dstl colleagues, and in particular Dr. Louise Martingale who was involved in the development and application of the method for initial assessment in 2014 and who has provided an independent technical review of this chapter.

References

1. Levene L (2011) Defence reform: an independent report into the structure and management of the Ministry of Defence. Stationery Office, Norwich
2. Lowe D, Martingale L, Yearworth M (2016) Guiding interventions in a multi-organisational context: combining the viable system model and hierarchical process Modelling for use as a problem structuring method. J Oper Res Soc 67(12):1481–1495
3. Delivering the Defence Estate, National Audit Office (2016)
4. Kotter JP (1996) Leading change
5. Checkland P (1981) Systems thinking, systems practice
6. Senge PM (1997) The fifth discipline. Meas Bus Excell 1(3):46–51
7. Poli R (2014) Anticipation: what about turning the human and social sciences upside down? Futures 64:15–18
8. http://www.macmillandictionary.com/dictionary/british/resilience. Accessed 30 Sep 2015
9. Holling CS (1973) Resilience and stability of ecological systems. Annu Rev Ecol Syst 4:1–23
10. Holling CS (1996) Engineering resilience versus ecological resilience. In: Engineering within ecological constraints. National Academy Press, Washington, D.C., pp 31–44
11. https://www.gov.uk/government/organisations/ministry-of-defence/about. Accessed 21 Sep 2016
12. Hollnagel E, Woods DD, Leveson N (2007) Resilience engineering: concepts and precepts. Ashgate Publishing, Ltd., Aldershot
13. Annarelli A, Nonino F (2015) Strategic and operational management of organizational resilience: current state of research and future directions. British Library Serials, Omega
14. Rosen J, Kineman JJ (2005) Anticipatory systems and time: a new look at Rosennean complexity. Syst Res Behav Sci 22(5):399–412
15. Rosen R (2012) Anticipatory systems. In: Anticipatory systems. p. 313–370
16. Keller S, Price C (2010) Beyond performance – how great enterprises build ultimate competitive advantage
17. Lowe D, Yearworth M (2016) Ensuring continued enterprise resilience: developing a method for monitoring health. In: Conference on systems engineering research
18. Eden C (1992) A framework for thinking about group decision support systems (GDSS). Group Decis Negot 1(3):199–218
19. Midgley G (2000) Systemic intervention: philosophy, methodology, and practice. Springer, Boston
20. Mingers J, Gill A (1997) Multimethodology: theory and practice of combining management science methodologies. Wiley, Chichester
21. Hall JW, Blockley DI, Davis JP (1998) Uncertain inference using interval probability theory. Int J Approx Reason 19(3–4):247–264
22. Marashi SE (2006) Managing discourse and uncertainty for decision-making in civil and infrastructure engineering systems. University of Bristol, Bristol
23. Beer S (1979) The heart of enterprise, vol 2. John Wiley & Sons, Chichester
24. Beer S (1981) Brain of the firm
25. Beer S (1985) Diagnosing the system for organizations
26. Midgley G et al (2013) Towards a new framework for evaluating systemic problem structuring methods. Eur J Oper Res 229(1):143–154

27. Pawson R, Tilley N (2004) Realistic evaluation
28. Franco LA (2009) Problem structuring methods as intervention tools: reflections from their use with multi-organisational teams. Omega Inter J Manag Sci 37(1):193–203
29. Taket AR, White L (2000) Partnership and participation: decision-making in the multiagency setting
30. Gregory W, Midgley G (2000) Planning for disaster: developing a multi-agency counselling service. J Oper Res Soc 51(3):278–290
31. Huxham C, Cropper S (1994) From many to one-and back. An exploration of some components of facilitation. Omega 22(1):1–11
32. Franco LA (2013) Rethinking soft OR interventions: models as boundary objects. Eur J Oper Res 231(3):720–733
33. Friend JK, Hickling A (2005) Planning under pressure: the strategic choice approach. Elsevier/ Butterworth-Heinemann, Amsterdam
34. Pöppelbuß J, Röglinger M (2011) What makes a useful maturity model? a framework of general design principles for maturity models and its demonstration in business process management. ECIS 2011 Proceedings. 28

Chapter 54
An Analysis of Individual Systems Thinking Elements

Susan Ferreira and Divya Behl

Abstract Systems thinking skills are important in engineering complex systems. Systems thinking includes approaches to understand and address a problem or need as well as potential solutions from holistic perspectives. Multiple studies identify key elements related to individual systems thinking. While the various studies identify sets of elements, there is limited information that defines these elements and relates them. Rubrics for these elements are also missing. In this paper, the authors discuss the identification of a set of individual systems thinking elements. The authors define the elements, establish rubrics, and discuss relationships between the various elements.

Keywords Systems thinking • Systems thinking rubrics

54.1 Introduction

Systems thinking is critical in successfully engineering complex systems. Systems thinking allows one to view the world as a complex system including understanding its connections and interrelationships [1]. Systems thinking offers a holistic view of a system and its context. It focuses on the "whole system."

As systems become more complex, the need for a workforce that has systems thinking skills that enable the conceptualization, development, and management of these complex systems is increasing. Systems thinking is an important competency for systems engineers. Systems thinking is included as a competency category in the International Council on Systems Engineering (INCOSE) Systems Engineering Competencies Framework [2]. Systems thinking is identified as a "backbone" of the Systems Engineering Experience Accelerator Competency Taxonomy [3].

Research related to systems thinking is important. INCOSE identifies in its Systems Engineering Vision 2025, "The research roadmap will mature systems

S. Ferreira (✉) • D. Behl
The University of Texas at Arlington, Systems Engineering Research Center, Arlington, TX, USA
e-mail: ferreira@uta.edu

A.M. Madni et al. (eds.), *Disciplinary Convergence in Systems Engineering Research*, https://doi.org/10.1007/978-3-319-62217-0_54

thinking and the theoretical foundation as a basis for advanced systems engineering methods and tools" [4].

While studies have been performed related to individual systems thinking elements, there is limited information that defines the individual elements and shows their relationships. An integrated set of rubrics or measures for these elements are also missing. Rubrics are required to provide a way of evaluating different levels of each element.

This paper first provides background information related to systems thinking and distinguishes between individual systems thinking and team systems thinking. The authors build upon previous research and present individual systems thinking elements, element definitions, and element rubrics. The authors also present a model that graphically illustrates relationships between the elements.

54.2 Individual Systems Thinking Background

The authors distinguish between individual systems thinking and team systems thinking [5, 6]. This distinction is important given the focus of the paper on individual systems thinking.

54.2.1 Individual Systems Thinking

Individual systems thinking (IST) is the ability of a specific engineer to exhibit systems thinking. Various studies identify the traits, elements, and characteristics that are required for IST. Heidi Davidz [7] identified characteristics and traits through the use of literature reviews, surveys, interviews, field studies, data analyses, and theory analyses. The field study consisted of 205 participants from 10 companies. It also included expert panelists, both senior and junior systems engineers, and technical specialists.

Moti Frank [8] conducted another study of individual systems thinking. Frank identified the capacity for engineering systems thinking (CEST) and a resulting CEST measurement tool. The tool utilizes an interest inventory that was developed using literature reviews, field studies, and observations. To validate and strengthen the study, Frank used a triangulation technique that compared the inventory list with the results of studies conducted by Frampton, Thom and Carroll [9], and Di-Carlo and Khoshnevis [10]. Frank's findings illustrated characteristics that were found in all three studies.

Stave and Hopper [11] developed a taxonomy that identifies seven characteristics of a systems thinker. These characteristics include recognizing interconnections, identifying feedback, understanding dynamic behavior, differentiating types of variable and flows, using conceptual models, creating simulation models, and testing policies. The taxonomy further defines the characteristics by including measures and objectives for each.

Moore et al. [12] developed the systems thinking scale (STS) as a way to measure the systems thinking capability needed by professionals in the health-care industry. Eleven experts in the fields of systems thinking and continuous improvement provided inputs to develop the results. The results were validated by health-care professionals and later further validated using psychometric testing. The resulting 30-question STS uses six dimensions of systems thinking: sequence of events, causal sequence, multiple causations possible, variation of different types, feedback, and interrelations of factors. Additional studies focused on individual systems thinking factors include Behl and Ferreira [5, 6], Arnold and Wade [13], and Camelia and Ferris [14].

54.2.2 Team Systems Thinking

Collaborative systems thinking (CST) as defined by Lamb [15] is "an emergent behavior of teams resulting from the interactions of team members." Lamb's research identified eight generalized traits of CST teams. Lamb's research concludes that CST teams: engage in more consensus decision making, have three categories of membership, have communication preferences for real-time group interactions, have a higher number of past and concurrent program experience, rate their environment favorably, have more creative environments, require both technical and social leadership, and are more likely to engage in CST. Collaborative systems thinking is identified as team systems thinking (TST) in this paper.

54.2.3 IST Element Research

The authors of this paper refer to the characteristics, elements, and traits from the different classifications as systems thinking "elements." In a previous paper, Behl and Ferreira [6] identified IST elements by comparing and contrasting the elements identified by Frank [8], Davidz [7], Stave and Hopper [11], and Moore et al. [12].

A set of guidelines was established to assist in identifying key IST elements. The guidelines considered the frequency of element occurrences in the survey responses from previous studies. The first criterion used in the analysis was to compare the complete set of elements from Frank's [8] research with the elements from Davidz [7], Stave and Hopper [11], and Moore et al. [12]. In order for an element to be included in the set of IST elements, it had to be first in Frank's [8] list of elements as well as in at least one of the other studies [7, 11, 12]. Frank's [8] systems thinking elements are used as the base in the first criterion because he validated his research against other researchers. He also provided background information for each element. As part of the first criterion, Davidz's [7] elements that had greater than or equal to five survey responses were included. A minimum requirement of five survey responses identifying an element was used to avoid cases where the survey question may have been interpreted incorrectly. A separate criterion included elements from Davidz's [7] research that had greater than or equal to ten responses, regardless of whether or not the element existed on Frank's [8] list. A quantity of ten was used in this case because it represents a reasonable quantity in relationship to the rest of the responses in the survey [6]. The application of the two criteria resulted in the identification of 21 IST elements. If the identified element name varied between the previous researchers, the authors of this paper adapted the element name to fit the needs most suitable to the focus of their research. The original set of 21 elements is presented in Behl and Ferreira [6].

54.3 IST Element Definitions and Rubrics

In order to continue the research, definitions and rubrics needed to be identified for the elements. As definitions and rubrics were developed for each of the elements, the set of elements was reduced to 18 elements from the original 21. The set of elements was reduced because as the elements were defined and rubrics were selected for each of the elements, it was determined that a number of elements were not sufficiently distinct from one another. Given this, the redundant elements were eliminated from the original set. Element names were also updated from those presented in Behl and Ferreira [6]. Table 54.1 presents the current set of IST elements.

Element definitions were developed using available information from the previous IST element researchers. However, in many cases the existing set of IST element definitions were very limited or did not exist. Elements were then adapted from other sources or defined by the authors. Table 54.2 shows a subset of the IST element definitions.

The authors conducted an extensive review of the literature on measures and measurement methods for each of the IST elements. The review was done using each of the respective element names as well as terms related to the rubrics. While several rubrics might exist for an IST element, the authors attempted to select the rubric best aligned with the element definition. Many of the rubrics required an adaptation to fit the use in the planned research. Table 54.3 shows an example of two rubrics identified for the "understanding interconnection" element. The first rubric from Plate [20] was selected for the understanding interconnection element. Table 54.4 shows a subset of other examples of the rubrics the authors selected for their system elements.

Table 54.1 Individual systems thinking elements

IST element name
Understanding the whole system
Understanding interconnections
Thinking creatively
Using multiple perspectives
Being curious
Asking good questions
Being analytical
Creating, building, and using models
Having good communication skills
Having self confidence
Being disciplined
Thinking abstractly
Having initiative/motivation
Having a systems engineering education
Having wide and varied experience
Being outgoing/extrovert
Having a tolerance for uncertainty
Having good listening skills

Table 54.2 IST element definition examples

IST element name	IST element definition
Understanding interconnections	Understanding interconnections means having the knowledge and ability to understand relationships and interdependencies between system elements at various hierarchical levels of the system along with the results of interactions between system elements (adapted from Webster [16], Stave and Hopper [11])
Using multiple perspectives	Using multiple perspectives means understanding the system from diverse and several points of view (adapted from Frampton, Thom, Carroll [9])
Asking good questions	Asking good questions means inquiring to elicit critical and/or key information (Behl and Ferreira [17])
Having wide and varied experience	Having a wide and varied experience means having many forms or types of experiences, knowledge, and education (adapted from Webster [18])
Understanding the whole system	Understanding the whole system means comprehending the system holistically, taking into consideration all its elements, subsystems, assemblies, and components and recognizing that the system is greater than the sum of its parts (adapted from Frank [8], Senge [19])
Having good communication skills	Having good communication skills means having exemplary oral and written skills (adapted from Frampton, Thom, and Carroll [9])

Table 54.3 IST element rubric option example

Rubric name	Source	Adaptation
Recognizing interconnection measure	Plate [20]	Adapted category names. The rest is used as is
Interdependencies	Waters Foundation [21]	

Table 54.4 IST element rubric examples

Rubric name	Source	Adaptation
Using multiple perspectives	Waters Foundation [21]	Adapted from big picture rubric
Asking good questions	Alberta Canada Government [22]	Changed category names. Used only the first term in the definition to keep the rubric simple
Having wide and varied experience	Behl and Ferreira [17]	Developed new rubric
Understanding the whole system	Waters Foundation [21]	Adapted material from page 2 (big picture), page 6 (big picture), and page 4 (big picture WOW bullet) from Waters Foundation
Having good communication skills	DoDEA [23]	Adapted the student portion of the rubric

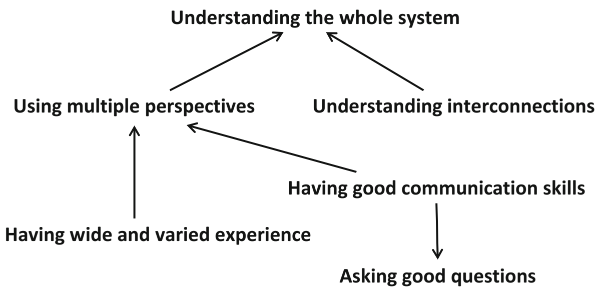
54.4 IST Element Relationships

The authors analyzed the relationships between the identified IST elements. Important relationships do exist between the IST elements where individual IST elements do contribute to other IST elements. Figure 54.1 illustrates a subset of the identified relationships between IST elements. Using multiple perspectives and understanding interconnections contribute to understanding the whole system. Having good communication skills and a wide and varied experience contributes to using multiple perspectives. Having good communication skills contributes to asking good questions.

54.5 Conclusions

This paper presents a set of IST elements distilled from previous research. Definitions and rubrics were developed for each of these elements. A subset of these is discussed in the paper. The rubrics are important because they provide a basis to measure each of the elements. The authors graphically present results of an analysis of relationships between a subset of the elements. The authors plan to utilize results from this research to explore the potential association between IST elements and other systems engineering factors such as project success.

Fig. 54.1 IST element relationships


Understanding the whole system
Using multiple perspectives
Understanding interconnections
Having good communication skills
Having wide and varied experience
Asking good questions

References

1. Sterman J (2000) Business dynamics: systems thinking and modeling for a complex world. McGraw-Hill Companies, Inc., Boston
2. International Council on Systems Engineering (INCOSE) (2010). Systems Engineering Competencies Framework, Document No.: INCOSE-TP-2010-003, Version/Revision 3. INCOSE UK Systems Engineering Competencies Working Group
3. Squires A, Wade J, Dominick P, Gelosh D (2011) Building a competency taxonomy to guide experience acceleration of lead program systems engineers. Conference on Systems Engineering Research, Redondo Beach
4. INCOSE (2014) INCOSE systems engineering vision 2025
5. Behl DV, Ferreira S (2014) An analysis of individual and team systems thinking. Conference on Systems Engineering Research, Redondo Beach
6. Behl DV, Ferreira S (2014) Systems thinking: an analysis of key factors and relationships. Complex adaptive systems conference, Philadelphia; Procedia Computer Science 36:104–109.
7. Davidz H (2006) Enabling systems thinking to accelerate the development of senior systems engineers. Doctoral Dissertation, Massachusetts Institute of Technology
8. Frank M (2010) Assessing the interest for systems engineering positions and other engineering positions' required capacity for engineering systems thinking (CEST). J Syst Eng 13:161–174
9. Frampton K, Thom JA, Carroll J. Enhancing IT architect capabilities: experiences within a university subject, Australasian conferences inform system (ACIS2006), Adelaide, South Australia, Australia, 6–8 December 2006
10. Di-Carlo T, Khoshnevis B (2006) Whole-brain thinking in systems architecting, conference systems engineering research. University of Southern California, Los Angeles
11. Stave K, Hopper M (2007) What constitutes systems thinking? A proposed taxonomy. Proceedings of the 26th international conference of the system dynamics society. Athens
12. Moore SM, Dolansky MA, Singh M, Palmieri P, Alemi P (2010) The systems thinking scale. Unpublished manuscript
13. Arnold RD, Wade JP (2014) A definition of systems thinking: a systems approach. Conference on Systems Engineering Research, Redondo Beach
14. Camelia F, Ferris TLJ (2016) Undergraduate students' engagement with systems thinking: results of a survey study. IEEE Transactions on Systems, Man, and Cybernetics: Systems. doi:10.1109/TSMC.2016.2563386:1-12
15. Lamb C. (2009) Collaborative systems thinking: an exploration of the mechanisms enabling team systems thinking. Doctoral Dissertation: Massachusetts Institute of Technology
16. Webster. http://www.merriam-webster.com/dictionary/understanding. Last accessed 3/29/2014
17. Behl DV, Ferreira S (2016) Individual systems thinking model. Unpublished manuscript
18. Webster. http://www.merriam-webster.com/dictionary/experience. Llast accessed 3/29/2014
19. Senge P (1990) The fifth discipline. Doubleday, New York

20. Plate R, Monroe M (2014) A structure for assessing systems thinking. Creat Learn Exch 23 (1):1. 3–6, 12
21. Waters Foundation (2007) CFSD 21st century learning rubric, skill: systems thinking
22. Alberta Canada Government. Teacher rubric for asking powerful questions. https://www.learnalberta.ca/content/ssblm/word/teacherrubricforaskingpowerfulquestions_blm.doc. Last accessed 2/1/2017
23. Department of Defense Education Activity (DoDEA). 21st century teaching, learning, and leading; 21st century skills, reflection and evaluation rubrics; effective oral and written communication. https://content.dodea.edu/teach_learn/professional_development/21/docs/21st_century_skills_rubrics/Learner%20Outcomes%20and%20Reflection%20and%20Evaluation%20Rubrics.pdf. Llast accessed 2/1/2017

Part VII
Systems Engineering and Decision Science

Chapter 55
Using Bayesian Networks to Validate Technology Readiness Assessments of Systems

Marc F. Austin, Cheyne Homberger, George A. Polacek, Erin Doolittle, Virginia Ahalt, and Donald M. York

Abstract Currently, Technology Readiness Assessments (TRAs) are used in determining the maturity of the Critical Technology Elements (CTEs) of a system as it moves forward in the system development life cycle. The TRA method uses Technology Readiness Levels (TRLs) as the decision metric. TRL values are assessed and determined by Subject Matter Experts (SMEs). Since expert evaluators often differ in their judgment when scoring a system element against the TRL scale criteria, this paper argues for the use of a Bayesian network model to provide a mathematical method to consistently validate the judgment of these SMEs and increase the confidence in the determination of the readiness of system components and their technologies.

Keywords Technology • Readiness • Assessment • Technology Readiness Assessment • TRA • Bayesian • Bayes • Network • Critical • Technology • Element • CTE • Critical Technology Element • Technology Readiness Level • TRL

55.1 Introduction

Bayesian probability provides a framework for building and refining models which incorporate uncertainty. A Bayesian network, or Bayes net, is a graphical representation of a multidimensional probability distribution, in which a variety of indicators may be dependent on a complex network of observable and hidden variables. Bayes nets are well suited for translating complex relationships of dependencies into intuitive and mathematical models, and perform well even in the face of missing or inconsistent data.

M.F. Austin (✉) • C. Homberger, PhD • G.A. Polacek, PhD • E. Doolittle • V. Ahalt
US DoD, Arlington, VA, USA
e-mail: mmfaustin@gmail.com; cheyneh@umbc.edu

D.M. York, PhD
TASC, an Engility Company, Chantilly, VA, USA
e-mail: donald.york@engilitycorp.com

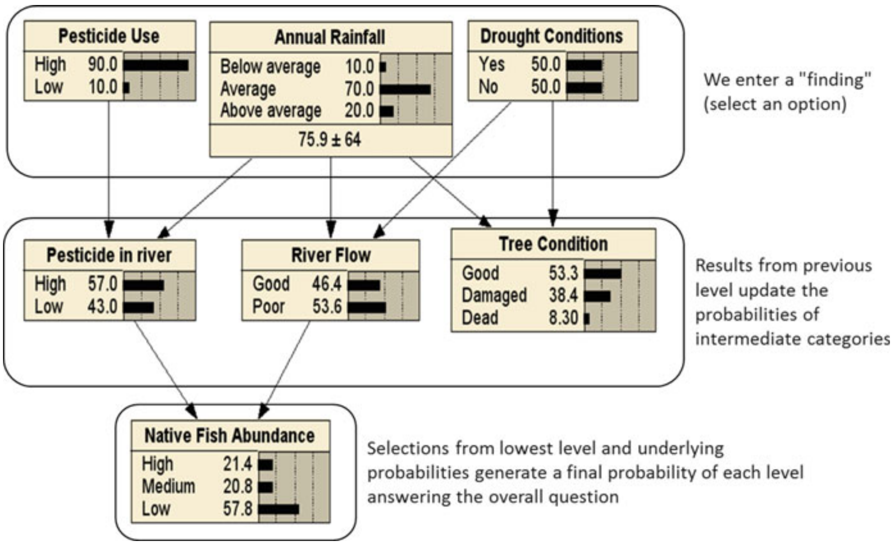


Fig. 55.1 An example of Bayesian belief network – predicting native fish abundance

In this case our challenge is the decision-making process that assigns Technology Readiness Level (TRL) values to system technologies or Critical Technology Elements (CTEs) in the Department of Defense (DoD)'s Technology Readiness Assessment (TRA) process. In performing a TRA, assessors consider a variety of different and often subjective attributes of a system in order to make a final determination which is as consistent as possible. The Bayes net effectively models this situation: it is able to incorporate a set of complex, possibly incomplete, and highly interrelated attributes and, through the laws of probability, produce a consistent and mathematically rigorous recommendation. The model is constructed through gathering evidence and eliciting expert opinion which are all incorporated, along with any uncertainty, in the final product. Each individual indicator is represented as a node in the network, with links representing dependencies between the nodes. Bayes Theorem governs the relationships between the connected nodes. Figure 55.1 illustrates an example of Bayesian belief network.

55.2 Why Use Bayesian Networks for TRLs?

The TRL is a systematic metric/measurement to assess the maturity of a particular technology and to allow consistent comparisons of maturity between different types of technologies. The TRL was initially pioneered by J.C. Mankins [1] at the National Aeronautics and Space Administration (NASA) Goddard Space Flight Center in the 1980s as a method to assess the readiness and risk of space technology. Over time, NASA continued to use readiness levels as part of an overall risk assessment process and as a means for comparison of maturity between various technologies. NASA incorporated the TRL methodology into the NASA Management Instruction 7100 as a systematic approach to the technology planning process. The DoD, along with several other organizations, later adopted this metric and tailored its definitions to meet their needs. TRL values range from 1 to 9. A definition, description, and decision criteria for TRL values 5, 6, and 7 are provided in Table 55.1. The Technology Readiness Assessment (TRA) method uses TRLs

Table 55.1 Decision criteria for assessing Technology Readiness Level (TRL)

TRL	Definition	Description	Supporting information
5	Component and/or breadboard validation in a relevant environment	Fidelity of breadboard technology increases significantly. The basic technological components are integrated with reasonably realistic supporting elements to they can be tested in a simulated environment. Examples include "high-fidelity" laboratory integration of components	Results from testing laboratory breadboard system are integrated with other supporting elements in a simulated operational environment. How does the "relevant environment" differ from the expected operational environment? How do the test results compare with expectations? What problems, if any, were encountered? Was the breadboard system refined to more nearly match the expected system goals?
6	System/subsystem model or prototype demonstration in a relevant environment	Representative model or prototype system, which is well beyond that of TRL 5, is tested in a relevant environment. Represents a major step up in a technology's demonstrated readiness. Examples include testing a prototype in a high-fidelity laboratory environment or in a simulated operational environment	Results from laboratory testing of a prototype system that is near the desired configuration in terms of performance, weight, and volume. How did the test environment differ from the operational environment? Who performed the tests? How did the test compare with expectations? What problems, if any, were encountered? What are/were the plans, options, or actions to resolve problems before moving to the next level?
7	System prototype demonstration in an operational environment	Prototype near or at planned operational system. Represents a major step up from TRL 6 by requiring demonstration of an actual system prototype in an operational environment (e.g., in an aircraft, in a vehicle, or in space)	Results from testing a prototype system in an operational environment. Who performed the tests? How did the test compare with expectations? What problems, if any, were encountered? What are/were the plans, options, or actions to resolve problems before moving to the next level?

Developed by NASA and recommended by the Defense Acquisition Guidebook and is a Department of Defense (DoD) directive performed across the DoD [2]. A TRL of 6 is a particularly critical milestone in the US DoD systems development life cycle as it is required to enter full scale development.

In the Technology Readiness Assessment, Critical Technology Elements or CTEs are selected from among the elements/components of the development system. The TRL of each of these CTEs is assessed by Subject Matter Experts (SMEs). Although scoring a technology in conjunction with the 1 to 9 TRL Scale is based on satisfying certain requirements and providing the accompanying evidence, expert evaluators may often differ in their judgment. Use of the Bayesian network and the resulting probability distributions serve to validate the judgment of these SMEs.

The Bayes net provides an effective framework for testing the most likely outcome of future events or scenarios and finding their likely cause(s). It combines both subjective expert opinions with available quantitative information/data providing informed decision-making without requiring complete knowledge of the problem. The problem domain experts take ownership because their input is vital. As new knowledge is acquired, the convenient modular design of the Bayes net accommodates this added information.

55.3 Constructing the Bayesian Network

Figure 55.2 illustrates the process steps followed in constructing the TRL Bayesian network. First, determine the main question of interest. Then identify the set of variables or nodes. Is there a natural ordering of these nodes? Can the nodes be treated as having binary states? Next, what is the dependency structure? Ascertain how to preserve an acyclic requirement and how to significantly reduce complexity. Finally, what is the Conditional Probability Table (CPT) for each Node?

There are certain requirements for a "good" variable. The values (or states) must be mutually exclusive. In other words, two or more states cannot be true at any single time. Also, states in a variable must be collectively exhaustive. One of the states must be true. Lastly, states must be unambiguously defined. Ambiguous states, such as "other", should be avoided. Use a "clarity" test for each variable. Clarity means that a person looking at the variable knows without interpretation or assumption what the variable means and what its value is.

Figure 55.3 shows the complete set of TRL "care-abouts" or variables that are eventually reduced to the lowest level nodes in the model. These variables resulted from a series of brainstorming meetings of the SMEs to "list" everything they thought contributed to or might potentially impact the decision-making process determining a TRL. While the conventional TRL tables define the TRL at each level, the purpose of the care-abouts is to capture a much more comprehensive and detailed set of attributes considered by the SME in their decision-making process. Along the way numerous variables were combined, de-duped, or determined not to influence the TRL decision and were eliminated. During this process, focus is

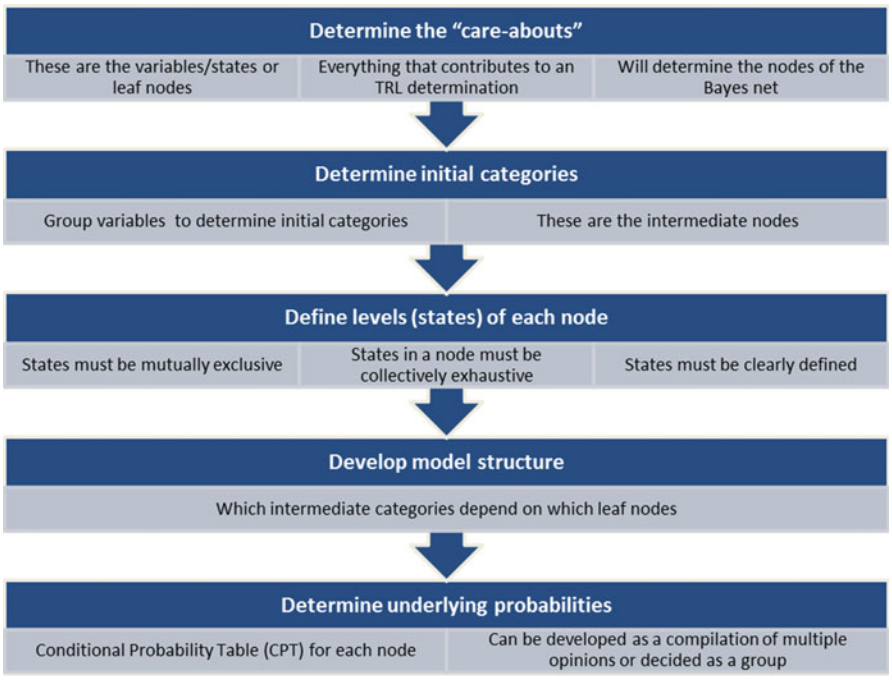

Determine the "care-abouts"
These are the variables/states or leaf nodes
Everything that contributes to an TRL determination
Will determine the nodes of the Bayes net
Determine initial categories
Group variables to determine initial categories
These are the intermediate nodes
Define levels (states) of each node
States must be mutually exclusive
States in a node must be collectively exhaustive
States must be clearly defined
Develop model structure
Which intermediate categories depend on which leaf nodes
Determine underlying probabilities
Conditional Probability Table (CPT) for each node
Can be developed as a compilation of multiple opinions or decided as a group

Fig. 55.2 Constructing a Bayesian network

placed on the "as is" condition, and factors that attempted to predict the future were intentionally removed from the initial set. Examples of the latter would be politics and sustainability.

The variables shown in Fig. 55.3 are then grouped into a set of initial "categories" which determine the intermediate nodes in the Bayesian belief network. For example, from the factors identified, a category that manifested itself early in the analysis was "verification." Figure 55.4 shows the initial set of care-abouts that were collected into this verification category. As will be seen later, the verification category was ultimately reduced to two subfactors, test environments and level of testing passed.

Once the initial categories are determined, the levels (states) of each node are clearly defined. States must be mutually exclusive. Figure 55.5 shows the main category knowledge and its three nodes or subcategories, research, proof-of-concept, and prior usage. Figure 55.4 reflects the progression of the development of the model structure as it portrays which intermediate categories depend on which leaf nodes.

The final step in the construction of the Bayesian network is to determine the underlying probabilities and construct a Conditional Probability Table (CPT) for each node. Table 55.2 provides some example entries in the CPT for the Impact of Technology Change Node. The interpretation of the table says, given the conditions

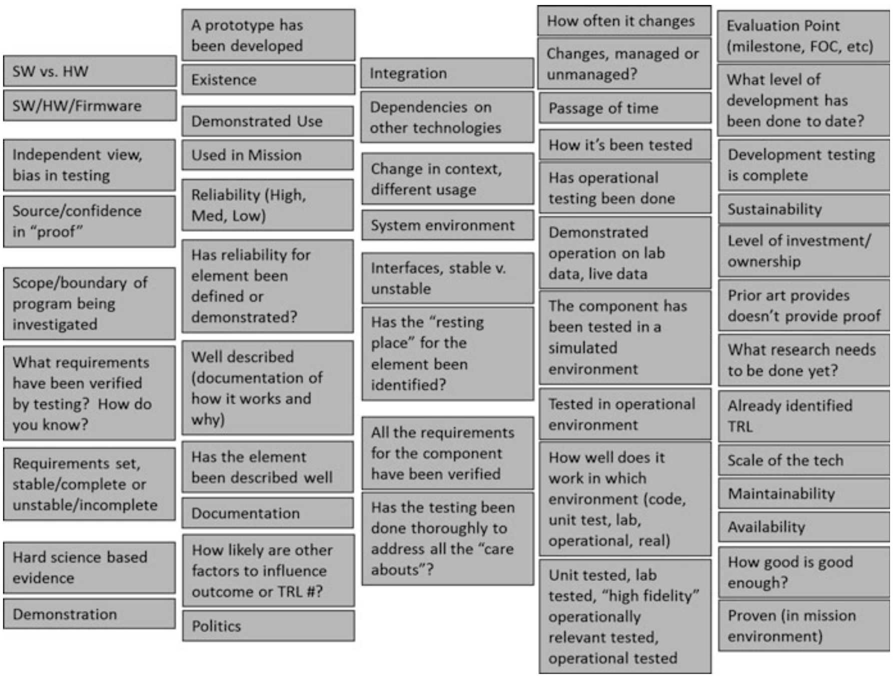

SW vs. HW
SW/HW/Firmware
Independent view, bias in testing
Source/confidence in "proof"
Scope/boundary of program being investigated
What requirements have been verified by testing? How do you know?
Requirements set, stable/complete or unstable/incomplete
Hard science based evidence
Demonstration
A prototype has been developed
Existence
Demonstrated Use
Used in Mission
Reliability (High, Med, Low)
Has reliability for element been defined or demonstrated?
Well described (documentation of how it works and why)
Has the element been described well
Documentation
How likely are other factors to influence outcome or TRL #?
Politics
Integration
Dependencies on other technologies
Change in context, different usage
System environment
Interfaces, stable v. unstable
Has the "resting place" for the element been identified?
All the requirements for the component have been verified
Has the testing been done thoroughly to address all the "care abouts"?
How often it changes
Changes, managed or unmanaged?
Passage of time
How it's been tested
Has operational testing been done
Demonstrated operation on lab data, live data
The component has been tested in a simulated environment
Tested in operational environment
How well does it work in which environment (code, unit test, lab, operational, real)
Unit tested, lab tested, "high fidelity" operationally relevant tested, operational tested
Evaluation Point (milestone, FOC, etc)
What level of development has been done to date?
Development testing is complete
Sustainability
Level of investment/ ownership
Prior art provides doesn't provide proof
What research needs to be done yet?
Already identified TRL
Scale of the tech
Maintainability
Availability
How good is good enough?
Proven (in mission environment)

Fig. 55.3 The initial set of variables or "care-abouts" from the TRL brainstorming meetings

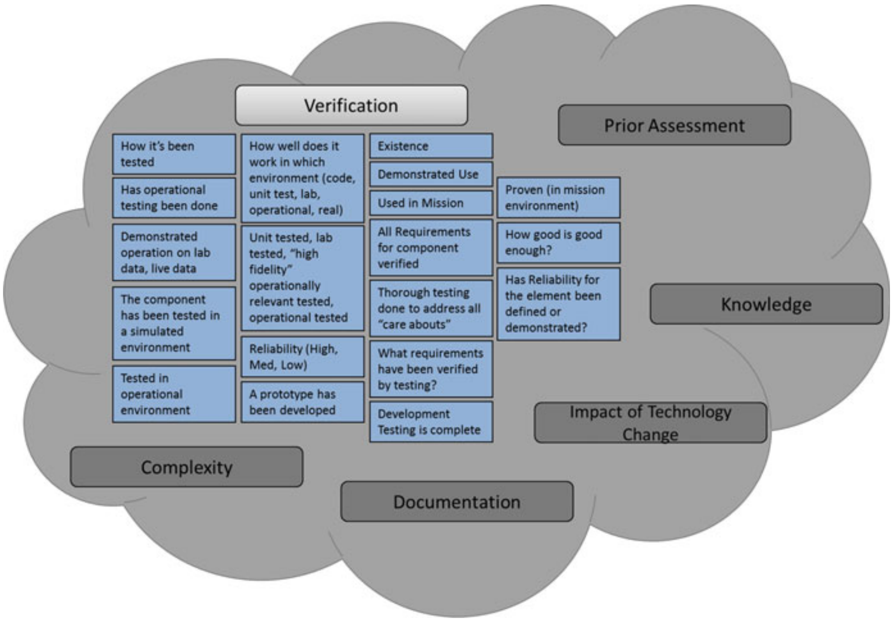

Verification
Prior Assessment
Knowledge
Impact of Technology Change
Complexity
Documentation
How it's been tested
Has operational testing been done
Demonstrated operation on lab data, live data
The component has been tested in a simulated environment
Tested in operational environment
How well does it work in which environment (code, unit test, lab, operational, real)
Unit tested, lab tested, "high fidelity" operationally relevant tested, operational tested
Reliability (High, Med, Low)
A prototype has been developed
Existence
Demonstrated Use
Used in Mission
All Requirements for component verified
Thorough testing done to address all "care abouts"
What requirements have been verified by testing?
Development Testing is complete
Proven (in mission environment)
How good is good enough?
Has Reliability for the element been defined or demonstrated?

Fig. 55.4 A snapshot of the Bayesian network categories and specifically the subcategories for verification

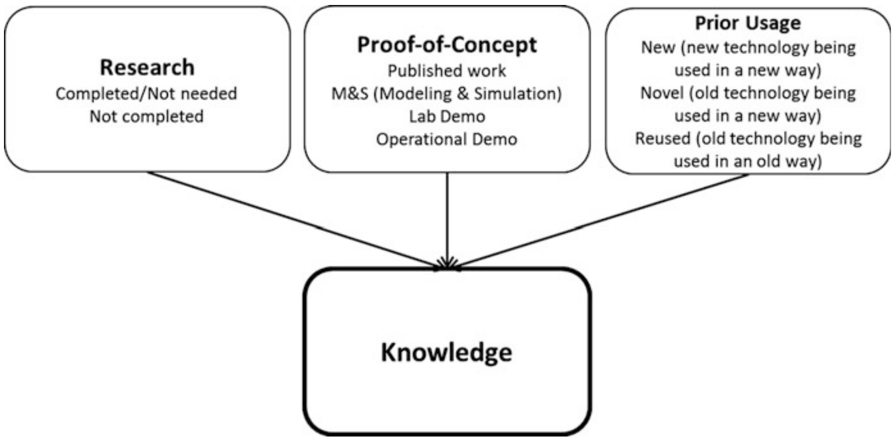

Research
Completed/Not needed
Not completed
Proof-of-Concept
Published work
M&S (Modeling & Simulation)
Lab Demo
Operational Demo
Prior Usage
New (new technology being used in a new way)
Novel (old technology being used in a new way)
Reused (old technology being used in an old way)
Knowledge

Fig. 55.5 The knowledge category and its subcategories for the Bayesian network model

Table 55.2 Sample Conditional Probability Table (CPT) for the Impact of Technology Change Node

Control of technology change	Frequency of technology change	Magnitude of technology change	Explanation	High	Moderate	Low
Managed	Frequent	Large	If managed technology change frequently occurs and the magnitude of that change is large, how likely is the impact of that technology change to be?	33	34	33
Managed	Never	Large	If managed technology change never occurs and the magnitude of that change is small, how likely is the impact of that technology change to be?	0	0	100
Unmanaged	Infrequent/ seldom	Small	If unmanaged technology change infrequently/seldom occurs and the magnitude of that change is small, how likely is the impact of that technology change to be?	60	30	10

in the first three columns, how likely is it that the state of the node (in this example, the Impact of Technology Change) will be characterized as high, moderate, or low?

The overall model is constructed by linking these individual CPTs together according to the structure of the model, culminating in the final TRL value. Each individual variable then has predictive power over the final result, but the influence is interdependent on the values of each of the other nodes. That is, a change to a single finding may have different results on the final TRL assessment depending on the states of the other nodes. For example, the quality of documentation may have a much stronger impact on the eventual assessment for an immature system than one which has already undergone testing.

55.4 Model Case Studies

Technology Readiness Assessments (TRAs) from existing or prior programs were sought in order to validate the Bayesian network model. The resulting questionnaire produced by the Bayesian network model is shown in Table 55.3. The category and subcategory headings provide context for the nodes. The questionnaire in Table 55.3 reflects a completed evaluation for a project whose TRL assessment was 3. No specific details are provided for any of the model case studies as the projects and systems involved were proprietary in nature. The following questionnaire instructions were given to the survey respondents:

Questionnaire Instructions We are exploring the use of a Bayesian network approach to enhance the current method of performing Technology Readiness Assessments (TRAs). The model uses the responses to the series of questions shown in the table that follows to ultimately generate a probability distribution of the TRL levels. Your feedback is important as it will be used to validate our model and assist us in determining the way forward.

Using the program response as input data, the Bayes net model produces the resulting TRL probability distribution shown in Fig. 55.6. As one can see the Bayes net model predicts about a 38% probability that the TRL $= 2$ and about a 30% probability that the TRL is 3. As was stated initially, the project had been assessed at a TRL of 3. One can also describe the results in terms of cumulative probability. In other words, there is approximately a 91% probability that the TRL is equal to or less than 3.

Another case study was provided where the system was assessed at a TRL of 7. Inputting the data from responses to the model questionnaire yielded the TRL probability distribution shown in Fig. 55.7. In this case it can be seen that the model indicates that there is approximately a 95% probability that the TRL is greater than 7. In both these case studies, evidence from the probabilities of the states of the nodes is compared with the TRL assessment done by Subject Matter Experts. In the first case study, the model indicates that the TRL is about 8% more likely equal to a value of 2 compared to the TRL of 3 assessed by the SMEs. In the second system,

Table 55.3 Bayesian network project questionnaire

Category	Subcategory	Question	Possible Answers (Select/highlight one)
Documentation	Quality	Correctness: How **accurate** is the documentation?	■ No Errors Found □ Only non-critical errors found □ Critical Errors Found
		How current: How **current** is the documentation?	□ Current ■ Somewhat current □ Not current
	Completeness	How **complete** is the documentation?	□ All Documentation Present ■ Missing only non-critical documents □ Missing critical documents □ No documentation present
Documentation includes e.g., acquisition documents, architecture products, engineering specs, test plans, and general references. **Critical errors** are those which cause a misunderstanding of the facts and significantly impact the outcome. A **critical document** is any document that contains data elements essential to understanding the technology under evaluation.			
Knowledge	Research	What is the status of the underlying research?	□ Completed/Not needed ■ Not completed
	Proof-of-Concept	To what level has the concept been demonstrated?	□ Published work ■ M&S (Modeling and Simulation) □ Lab Demo □ Operational Demo
	Prior Usage	How has it been used in the past?	■ New (new technology being used in a new way) □ Novel (old technology being used in a new way) □ Reused (old technology being used in an old way)
Prior Assessment	Historical TRL	Was it previously assessed at a certain TRL level? If so, what level?	□ ___2__ (Insert previous TRL 1-9 here, or leave blank if not previously assessed)
	Context Change	Has the context significantly changed (from prior assessment)?	■ Yes or Unknown (or, not previously assessed) □ No
Context change must be linked to a prior TRL assessment. If no prior assessment occurred, select "Yes or Unknown."			
Impact of Technology Change	Magnitude	What is the magnitude of technology change?	■ Large □ Small
	Frequency	What is the frequency of technology change?	□ Frequent ■ Infrequent/Seldom □ Never
	Control	What is the control of technology change? (How well is change controlled?)	■ Managed □ Unmanaged
Complexity	Scale	What is the scale?	■ Large □ Moderate □ Small
	Intra-dependencies	How many intra-dependencies?	■ Many □ Moderate □ Few

Table 55.3 (continued)

	Interdependencies	How many interdependencies?	■ Many ☐ Moderate ☐ Few
Scale refers to, for example, the scope, magnitude, quantity, or breadth of the technology within the system. **Intra-dependencies** are within the technology. **Interdependencies** are between the technologies.			
Verification	Test Environments	What is the environment in which the testing was conducted?	☐ Analytic ☐ Lab ■ Relevant ☐ Operational
	Level of Testing Passed	What level of testing has been passed?	■ None ☐ Unit/Component Testing ☐ Integration testing ☐ Acceptance testing ☐ Operational testing
Level of testing refers to the highest level of testing that has been fully completed and successfully passed, with accompanying evidence. Testing need not be comprehensive to be completed.			

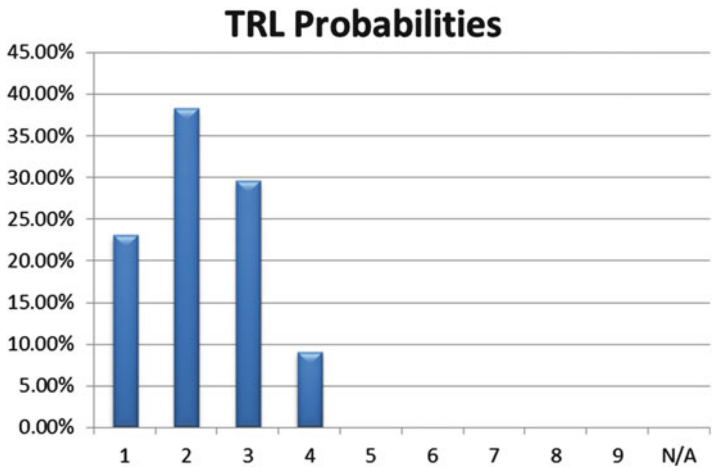


Fig. 55.6 Bayesian network model results for a case study where the assessed TRL = 3

the model shows a TRL assessed at two levels higher than what the SMEs concluded. In addition to the two case studies illustrated here, the model has been used with other systems and validation work is ongoing.

In summary, the use of the Bayesian network model provides the systems engineer with a level of confidence in the judgments made by the SMEs in assigning a TRL, mitigating the life cycle risk of the system components and their technologies.

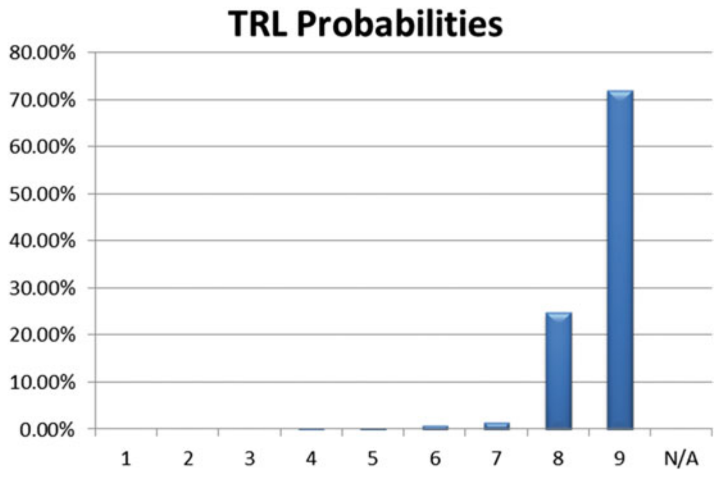


Fig. 55.7 Bayesian network model results for a case study where the assessed TRL = 7

55.5 Future Work

Work is currently being done to construct a Bayesian network model for Integration Readiness Levels (IRLs). The IRL is a metric to measure the integration maturity between two or more components. IRLs, in conjunction with TRLs, form the basis for the System Readiness Level (SRL), a systems level metric generated from the Systems Readiness Assessment (SRA) process [3]. The IRL values range from 0 to 9. The original IRL scale definitions, as proposed by Sauser [4], have been modified to be consistent with the foundation of the TRL scale and to reflect more closely our development model. IRLs represent the systematic analysis of the interactions between various components and provide a consistent comparison of the maturity between integration points. IRLs provide a means to reduce the risk involved in maturing and integrating components into a system. Similar to the TRL, the use of a Bayesian network model for IRLs will provide a mathematical method to consistently combine and validate the judgment of experts in the determination of the integration readiness of system components.

References

1. Mankins JC (1995) Technology Readiness Levels, NASA
2. Assistant Secretary of Defense for Research and Engineering (ASD(R&E)), Department of Defense Technology Readiness Assessment (TRA) Guidance, (April 2011)
3. Austin MF, York DM "System Readiness Assessment (SRA) – An illustrative example". In: 13th Annual Conference on Systems Engineering Research (CSER), Hoboken, NJ, 18–20 March 2015
4. Sauser B, Ramirez-Marques J, Magnaye R, Tan W (2008) A systems approach to expanding the technology readiness level within Defense acquisition. Inter J Defense Acqu Manag 1:39–58

Chapter 56
Adaptive and Automated Reasoning for Autonomous System Resilience in Uncertain Worlds

Curtis J. Marshalla, Blake Roberts, and Michael Grenn

Abstract As autonomous systems are increasingly employed in high-criticality applications, their safe and reliable operation is an overarching concern. Existing applications rely on comprehensive system characterization and preplanned contingencies for operating within known and anticipated circumstances. Human intervention is often required to supplement autonomous systems, but is less desirable as system complexity increases and infeasible for certain applications. Self-managing systems are sought to provide reliable and fully autonomous operation in virtually all circumstances, including hidden and emergent scenarios. Existing approaches to this challenge (i.e., rule-based and utility-based adaptation) lack validity in operational contexts with insufficient a priori knowledge or high uncertainty (which the authors define as uncertain worlds). This paper presents an adaptive and automated decision engine for improving autonomous systems' inherent resilience and adaptability. The concept is applied to FAA Free Flight initiatives to enhance air traffic management (ATM) safety and flexibility using increased automated technologies. The paper concludes with a discussion of an agent-based model (ABM) in development to evaluate the concept against leading methods for autonomous vehicle navigation and dynamic planning based on (a) mean success rate, (b) mean planning time, (c) mean time between policy violations, and (d) mean time to failure.

Keywords System resilience • Complex adaptive systems • Decision science • Autonomous aircraft navigation

C.J. Marshalla (✉) • B. Roberts • M. Grenn
George Washington University, Washington, DC, USA
e-mail: gwmail@gwu.edu; grennm@gwu.edu

56.1 Introduction

As the complexity of engineered systems and operating environments increase, artificial intelligence (AI) and autonomous functionality are becoming commonplace elements for performance monitoring and control. However, these capabilities do not inherently guarantee acceptable system performance in all operational contexts. Rather, these capabilities support varying degrees of system resilience – the ability of a system to manage disruptions and emergent needs. As defined in this paper, disruptions and emergent needs include anticipated, unexpected, inadvertent, and intentional occurrences and can originate internally or externally from the system. As explained by Madni and Jackson [1], resilience engineering extends beyond protection against system failures to any unacceptable performance degradation or perturbation. Additionally, Sood [2] emphasizes that the overarching goal in engineering resilient systems is to increase the likelihood of a system thriving, not just surviving, when disruptions are encountered. Inherent to this goal is the challenge of enabling sufficient system resilience in uncertain worlds (defined by the authors as operational contexts with insufficient a priori knowledge available and/or high uncertainty). Human intervention is often relied on as fail-safe or contingency for unsatisfactory performance in these circumstances. However, as system complexity increases, human support often becomes ineffective or infeasible for many applications. Given these constraints, self-managing systems providing reliable and fully autonomous operation are desired.

56.1.1 Research Challenge

Figure 56.1 illustrates the current and emerging paradigms for the use of AI and autonomous functionality. Given the lack of complete a priori knowledge of system risks and opportunities before deployment, high flexibility and robustness are desired to manage challenges in situ. While complete coverage of all unrecognized and emergent needs is infeasible, increased system reliability and availability in unforeseen circumstances is targeted. Optimization of autonomous system self-management processes under uncertainty is an inherent challenge toward this goal.

56.1.2 Motivations

Increasing change and uncertainty on local and global scales represent risks for realization and sustainment of engineered systems. Resilience engineering facilitates the proactive management of risks without explicit foresight. The following context is provided as real-world motivations for engineering resilient autonomous systems:

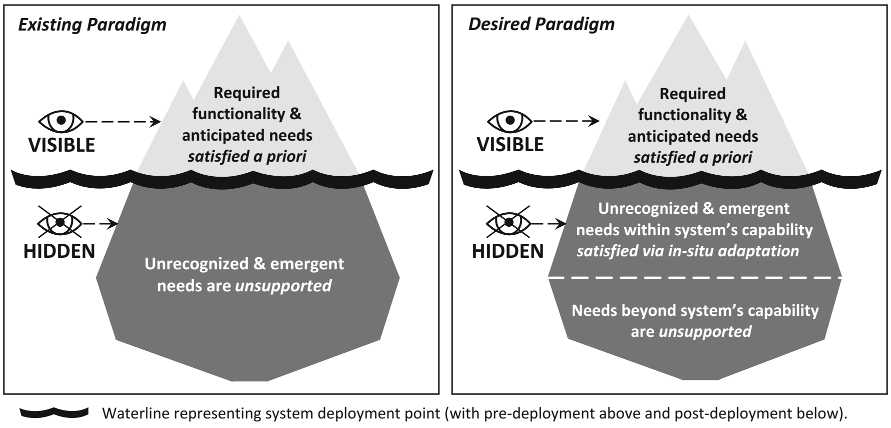

Existing Paradigm
VISIBLE
Required functionality & anticipated needs *satisfied a priori*
HIDDEN
Unrecognized & emergent needs are *unsupported*
Desired Paradigm
VISIBLE
Required functionality & anticipated needs *satisfied a priori*
HIDDEN
Unrecognized & emergent needs within system's capability *satisfied via in-situ adaptation*
Needs beyond system's capability are *unsupported*
Waterline representing system deployment point (with pre-deployment above and post-deployment below).

Fig. 56.1 Comparison of the existing and desired paradigms for artificial intelligence and autonomy using the Iceberg metaphor

- Increasingly dynamic and complex operating environments challenge human operators to provide effective monitoring and control to maintain acceptable system performance.
- Continuity of Operations (COOP) planning and emergency preparedness have become increasingly important due to the growing variety and volume of natural and man-made threats and vulnerabilities.

An underlying theme of these motivations is the need for high system reliability and availability in high-criticality systems and critical infrastructures. The implications of system failure or malfunction in these applications can be catastrophic, with detrimental impacts to public welfare including the potential loss of life. Resilient design and architecting have been championed in systems engineering research and practice as enablers of safe, secure, and trusted systems [1, 3, 4].

The uncertainty and complexity of air traffic management (ATM) exemplify the need for system resilience and adaptability [5]. ATM challenges have been exacerbated by the increase of unmanned aircraft systems (UAS) used for military, commercial, and private/hobbyist applications [6]. In the USA, over 540,000 small UAS (up to 55lbs) are expected to be in service by 2020, compared to an estimated 7200 aircraft for commercial aviation operations (for passenger and cargo transportation) [7]. As the demand for airspace increases, the role of autonomous technologies and procedures has also increased to improve ATM safety, flexibility, and efficiency. FAA Free Flight initiatives (Fig. 56.2) aim to improve every phase of flight with application to the aircraft and overarching ATM infrastructure [8, 9]. For example, the Surface Movement Advisory (SMA) capability supports increased efficiency in aircraft ground movement before takeoff and after landing. As the required capacity and complexity of ATM systems increases, fully autonomous solutions are sought to sustain and expand this vision.

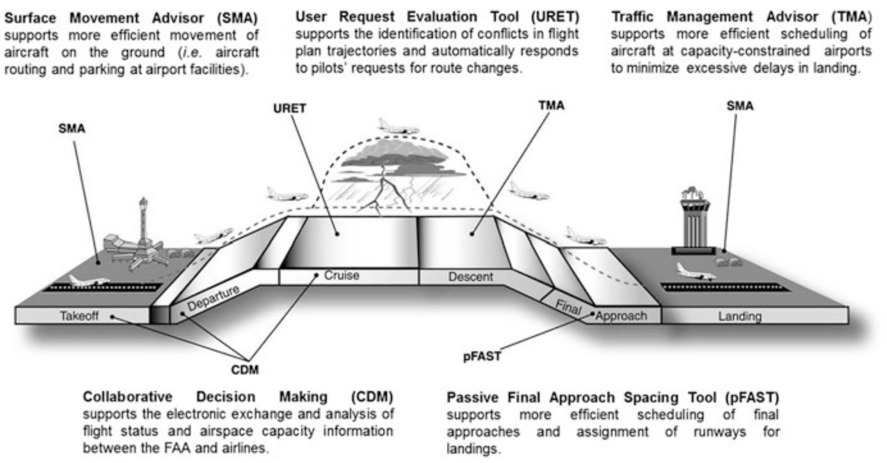

Surface Movement Advisor (SMA) supports more efficient movement of aircraft on the ground (i.e. aircraft routing and parking at airport facilities).
User Request Evaluation Tool (URET) supports the identification of conflicts in flight plan trajectories and automatically responds to pilots' requests for route changes.
Traffic Management Advisor (TMA) supports more efficient scheduling of aircraft at capacity-constrained airports to minimize excessive delays in landing.
SMA
URET
TMA
SMA
Takeoff
Departure
Cruise
Descent
Final
Approach
Landing
CDM
pFAST
Collaborative Decision Making (CDM) supports the electronic exchange and analysis of flight status and airspace capacity information between the FAA and airlines.
Passive Final Approach Spacing Tool (pFAST) supports more efficient scheduling of final approaches and assignment of runways for landings.

Fig. 56.2 Overview of FAA Free Fight (Phase I) initiatives to increase safety, flexibility, and efficiency of air traffic management (ATM) using increased autonomous technologies and procedures [8, 9]

56.2 Background

Resilience engineering focuses on the optimization of four principal attributes: capacity, flexibility, tolerance, and cohesion [10, 11] (defined in Table 56.1). Each attribute has applicability to physical and logical elements of a system. With respect to physical resilience, redundant and modular hardware are common measures taken to respectively increase system tolerance and flexibility. System software and communication features are often the target of measures to provide increased flexibility, tolerance, and cohesion.

56.2.1 Existing Approaches

The MAPE-K control loop (Fig. 56.3) is representative of conventional logical architecture for self-managing and self-adaptive systems enabling resilience. The model highlights self-assessment and self-regulation processes bridged by a shared knowledge base to enable system adaptations. Existing approaches for adaptation decision analysis and control logic can be generally classified into four categories: rule-based, model-based, utility-based, and learning-based (described in Table 56.2). Rule-based adaptation lacks flexibility but can offer excellent responsiveness for anticipated scenarios. On the other end of the spectrum, learning-based approaches enable high adaptability, but often suffer from poor responsiveness due to required training and/or calibration periods. Utility-based approaches offer a compromise between adaptability and responsiveness and are employed extensively in AI and autonomous systems. Hybrid strategies integrating multiple approaches can also be employed to achieve desired functionality.

Table 56.1 Overview of the core enabling attributes of system resilience

Resilience attribute	Description
Capacity	Enabling a system to withstand and/or resist disruption
Flexibility	Enabling the system to be reconfigured or adjusted before, during, and after a disruption
Tolerance	Enables a system to degrade gracefully and counteract cascading failures between related components/subsystems
Cohesion	Enabling system elements to communicate, cooperate, and collaborate with each other to avoid or recover from a disruption

Adapted from [10, 11]

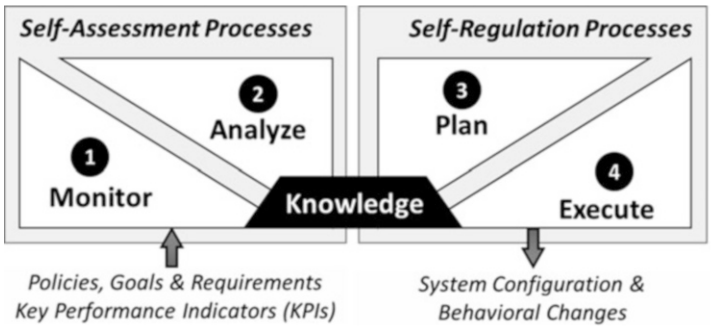

Self-Assessment Processes
Self-Regulation Processes
1 Monitor
2 Analyze
3 Plan
4 Execute
Knowledge
Policies, Goals & Requirements
Key Performance Indicators (KPIs)
System Configuration &
Behavioral Changes

Fig. 56.3 The MAPE-K reference model for self-adaptive systems (SAS) (Adapted from [12])

Table 56.2 Overview of adaptation approaches and strategies for self-managing systems

Adaptation approach	Description
Rule-based	Automatic system responses are triggered based on preestablished rules and policies. In circumstances without an applicable rule or policy, no change in system configuration or behavior is triggered
Utility-based	Adaptation alternatives are evaluated based on their anticipated value and/or benefit toward meeting system goals and objectives. Adaptation alternatives providing the most value or benefit drive system adaptations
Model-based	The system and operating environment are abstracted as models which are used to provide prognostic insight for adaptations
Learning-based	Patterns and trends of system behavior and performance are identified and exploited to support operational goals and objectives

Adapted from [13, 14]

56.2.2 *Emerging Approaches and Trends*

Significant efforts toward ethical decision-making using artificial agents are being pursued to supplement or replace human judgment [15, 16]. This research is concentrated on the assurance of authority, intent, and due diligence in decisions for applications with high criticality (e.g., weapons, safety systems, medical devices). Mimetic approaches (i.e., nature imitating or nature inspired) such as genetic algorithms and neural networks have also experienced significant growth [17]. These efforts are evidence of growing interdisciplinary focus and synergy surrounding AI and autonomous systems. In addition to traditional domains such as computer science and operations research, increased participation from the systems engineering, decision science, and public policy domains is evident. The pursuit of adaptive and scalable autonomy driven by situational context is a major theme among these efforts. An intrinsic element of this goal is the notion of a variable level of automation (LOA) – the degree of authority autonomous functionality has over system control [18]. The LOA spectrum ranges from entirely manual (i.e., no automation) to entirely autonomous system operation at opposite ends. However, a wide variety of semiautonomous schemas employing artificial agents with a human-in-the-loop (HITL) exist between these extremes. As such, the pursuit of alternatives to a HITL for semiautonomous and prospective autonomous applications is an enduring trend.

56.3 Targeted Contributions

The concept of *context adaptation* seeks to drive changes in system configuration and behavior based on situational awareness and operational context. Realizing this behavior requires the use of *bounded rationality (*i.e., constraint-based decision-making) given imperfect decision scenarios (i.e., inaccurate or incomplete data and/or resources to support the decision) [19, 20]. Bounded rationality also supports multiple forms of rationality (i.e., logical viewpoints or perspectives) and their concurrent application in the decision process. According to Sauter [21], there are six forms of rationality that support a rational and reasonable decision process (including economical, procedural, political, social/ethical, legal, and technical). However, all forms of rationality may not be required or available for all decisions. The overarching intent of context adaptation is to enable effective and efficient decision-making given these nuances. To enable this behavior, prioritization of the following characteristics lacking in existing decision strategies is targeted:

- Satisficing behavior – achieved when requirements are adequately and sufficiently satisfied given inherent constraints to determining or achieving optimal behavior [19, 20]. This contrasts with the prioritization of optimizing behavior sought in existing decision strategies.
- Anytime properties – enable a system to respond in a valid manner if disrupted or prevented from completing a task (i.e., expiration of time, lack of data, or other resources) [22, 23]. This contrasts with the uninterruptible and fault intolerant characteristics of many existing approaches.

Using a context-adaptation strategy, we seek improvement of the following measures of effectiveness and metrics related to system resilience: (1) mean success

Table 56.3 Summary of metrics and measures of effectiveness targeted for improvement

Performance measure/ metric	Description
Mean success rate (MSR)	Expected proportion of effective and timely adaptations in response to disruptions or emergent needs
Mean planning time (MPT)	Expected time required to identify or generate a successful adaptation in response to disruptions or emergent needs
Mean time between violations (MTBV)	Expected time between adaptations that violate system requirements and/or goals without resulting in system or mission failure
Mean time to failure (MTTF)	Expected time until system or mission failure due to an ineffective and/or untimely adaptation decision

rate (MSR), (2) mean planning time (MPT), (3) mean time between violations (MTBV), and (4) mean time to failure (MTTF) (defined in Table 56.3).

56.3.1 *Preliminary Hypotheses*

We present our preliminary null hypotheses (H_0) and alternative hypotheses (H_a) for each metric in Table 56.4. Subscripts denote the respective approaches for each metric (i.e., MSR_{CA}, MSR_{RB}, MSR_{UB}, MSR_{MB}, and MSR_{LB} indicate the mean success rate for context-adaptation, rule-based, utility-based, model-based, and learning-based approaches, respectively). A context-adaptation strategy is expected to increase MSR, MTBV and MTTF, while MPT is expected to decrease in comparison to existing adaptation approaches.

We base our hypotheses on context adaptation's potential to exploit suboptimal performance to meet system goals and objectives, whereas existing strategies inhibit or bias against this behavior. By prioritizing optimal behavior in an uncertain world, we believe existing approaches will forego satisficing opportunities to the detriment of the overall performance. As shown in Fig. 56.4, paradoxical behavior can occur in the pursuit of optimization. Figure 56.4a illustrates a quality paradox in which suboptimal global behavior can be achieved despite applying optimal local behaviors in the decision process (i.e., a greedy strategy). Figure 56.4b demonstrates an efficiency paradox in which poor resource utilization can be realized when analyzing alternatives with minimal and/or insignificant differences. While these examples represent quantitatively based decisions, these paradoxes also apply to decisions made on a qualitative basis. In the context of decision analysis and planning processes, autonomous or otherwise, these behaviors negatively impact quality (i.e., success rate), efficiency (i.e., planning time), and reliability (i.e., req. violation rates, failure rates).

These paradoxes of optimization can be explained, in part, by the concept of a contract algorithm – a process guaranteed to produce results of acceptable quality only when stipulated conditions are satisfied (e.g., data inputs, processing time) [22, 24]. Outside of these conditions, the process may yield unsatisfactory results or

Table 56.4 Preliminary research hypotheses

Performance measure/ metric	Context (CA) vs. rule-based (RB) adaptation	Context (CA) vs. utility-based (UB) adaptation	Context (CA) vs. model-based (MB) adaptation	Context (CA) vs. learning-based (LB) adaptation
Mean success rate (MSR)	H_0: $MSR_{CA} \leq MSR_{RB}$ H_a: $MSR_{CA} > MSR_{RB}$	H_0: $MSR_{CA} \leq MSR_{UB}$ H_a: $MSR_{CA} > MSR_{UB}$	H_0: $MSR_{CA} \leq MSR_{MB}$ H_a: $MSR_{CA} > MSR_{MB}$	H_0: $MSR_{CA} \leq MSR_{LB}$ H_a: $MSR_{CA} > MSR_{LB}$
Mean planning time (MPT)	H_0: $MPT_{CA} \geq MPT_{RB}$ H_a: $MPT_{CA} < MPT_{RB}$	H_0: $MPT_{CA} \geq MPT_{UB}$ H_a: $MPT_{CA} < MPT_{UB}$	H_0: $MPT_{CA} \geq MPT_{MB}$ H_a: $MPT_{CA} < MPT_{MB}$	H_0: $MPT_{CA} \geq MPT_{LB}$ H_a: $MPT_{CA} < MPT_{LB}$
Mean time between violations (MTBV)	H_0: $MTBV_{CA} \leq MTBV_{RB}$ H_a: $MTBV_{CA} > MTBV_{RB}$	H_0: $MTBV_{CA} \leq MTBV_{UB}$ H_a: $MTBV_{CA} > MTBV_{UB}$	H_0: $MTBV_{CA} \leq MTBV_{MB}$ H_a: $MTBV_{CA} > MTBV_{MB}$	H_0: $MTBV_{CA} \leq MTBV_{LB}$ H_a: $MTBV_{CA} > MTBV_{LB}$
Mean time to failure (MTTF)	H_0: $MTTF_{CA} \leq MTTF_{RB}$ H_a: $MTTF_{CA} > MTTF_{RB}$	H_0: $MTTF_{CA} \leq MTTF_{UB}$ H_a: $MTTF_{CA} > MTTF_{UB}$	H_0: $MTTF_{CA} \leq MTTF_{MB}$ H_a: $MTTF_{CA} > MTTF_{MB}$	H_0: $MTTF_{CA} \leq MTTF_{LB}$ H_a: $MTTF_{CA} > MTTF_{LB}$

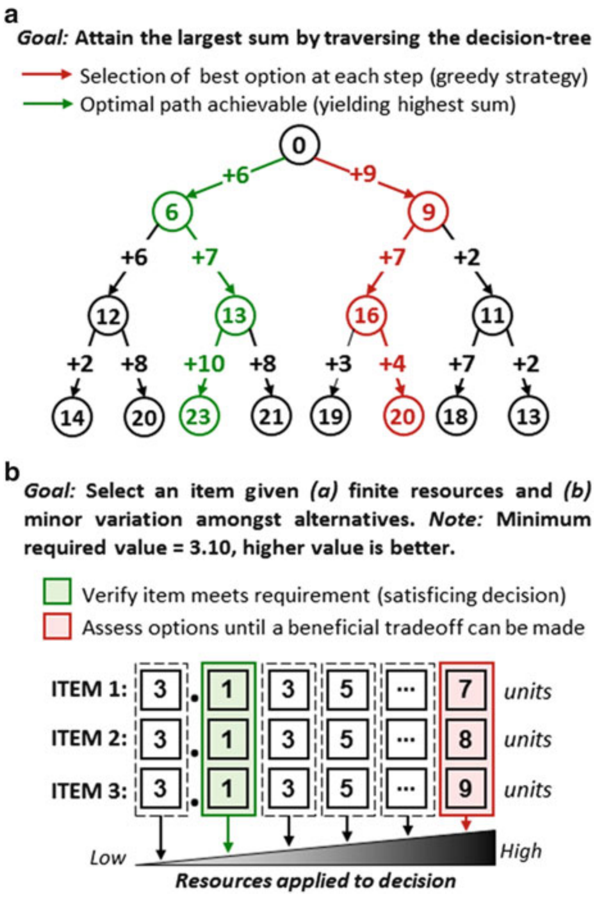

a
Goal: Attain the largest sum by traversing the decision-tree
Selection of best option at each step (greedy strategy)
Optimal path achievable (yielding highest sum)
0
+6
+9
6
9
+6
+7
+7
+2
12
13
16
11
+2
+8
+10
+8
+3
+4
+7
+2
14
20
23
21
19
20
18
13
b
Goal: Select an item given (a) finite resources and (b) minor variation amongst alternatives. Note: Minimum required value = 3.10, higher value is better.
Verify item meets requirement (satisficing decision)
Assess options until a beneficial tradeoff can be made
ITEM 1: 3 . 1 3 5 ··· 7 units
ITEM 2: 3 . 1 3 5 ··· 8 units
ITEM 3: 3 . 1 3 5 ··· 9 units
Low
High
Resources applied to decision

Fig. 56.4 (**a**) shows a quality paradox given the highest overall sum possible is not reached despite selecting the maximum value at each node; (**b**) shows an efficiency paradox given that an item can be selected without making definitive tradeoffs, requiring fewer resources for the decision

fail to provide any output. Existing adaptation strategies are representative of this approach given their generally predetermined and static logic for decision-making (i.e., rules, heuristics, ‘go-no go’ criterion, statistical correlation/confidence). In contrast, an interruptible algorithm [24] is designed to produce results of acceptable quality at any time if unexpected conditions are encountered during the process. This approach supports greater flexibility and robustness of a given process to inherent variability and uncertainty. These characteristics align with the adaptive and improvisational behavior sought via context adaptation. We believe these interruptible properties and the potential to exploit suboptimal behavior provide context adaptation and provide inherent advantages over existing approaches.

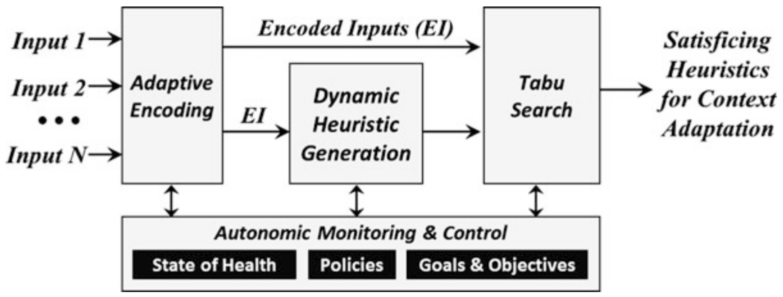


Fig. 56.5 Proposed decision engine for context adaptation in autonomous systems

56.4 Proposed Approach

An automated and adaptive decision engine is proposed to enable context adaptation in autonomous systems. Figure 56.5 illustrates the concept, highlighting the integration of four core functions and processes, including:

- Adaptive encoding – a data encoding schema and parsing convention to flexibly characterize inputs for use in a dynamic and indefinite decision process
- Dynamic heuristic generation – the creation or modification of decision heuristics in real time based on situational awareness and operational context (i.e., dynamic heuristics) [25]
- Tabu search – constraint-based search of the problem space and/or solution space [26, 27]
- Autonomic monitoring and control – self-assessment and self-regulation mechanisms to satisfice system operating policies, goals, and objectives [12, 28]

56.4.1 Target Application Area

Application of the concept to collision avoidance systems for autonomous aircraft is targeted. This application supports ATM modernization challenges and FAA Free Flight initiatives (discussed in Sect. 56.1.2). As shown in Fig. 56.6, inaccuracies in obstacle position (i.e., latitude, longitude, and altitude) introduce uncertainty into the collision avoidance scenario.

Constraints related to operational and meteorological conditions may also impact decision criteria and applicable approaches for collision avoidance. For example, Fig. 56.7 presents the two generally applicable situational contexts for aircraft collision avoidance in FAA Class G airspace. Figure 56.7a depicts collision avoidance by direct line of sight (LOS) given sufficient clearance from potential obstructions and visibility of the surrounding environment. Figure 56.7b illustrates the contingency to flight by LOS enabled by navigational aids and other visually indirect methods to detect obstacles. In both contexts, the “see and avoid” principle

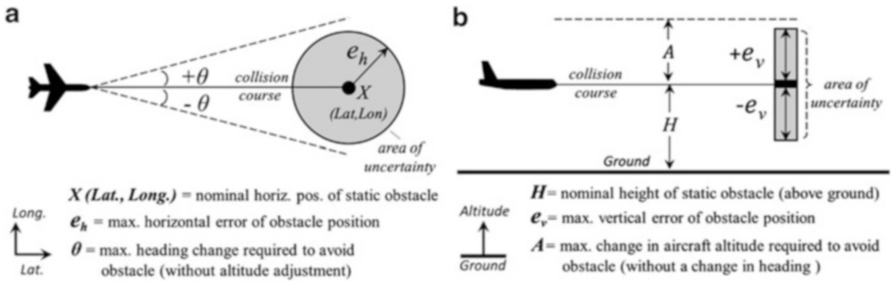

a
+θ
-θ
collision course
e_h
X
(Lat,Lon)
area of uncertainty
X (Lat., Long.) = nominal horiz. pos. of static obstacle
e_h = max. horizontal error of obstacle position
θ = max. heading change required to avoid obstacle (without altitude adjustment)
Long.
Lat.
b
collision course
A
H
$+e_v$
$-e_v$
area of uncertainty
Ground
H = nominal height of static obstacle (above ground)
e_v = max. vertical error of obstacle position
A = max. change in aircraft altitude required to avoid obstacle (without a change in heading)
Altitude
Ground

Fig. 56.6 (**a**) Horizontal error contributing uncertainty in aircraft collision avoidance with a static obstacle (**b**) Vertical error contributing to uncertainty in aircraft collision with a static obstacle

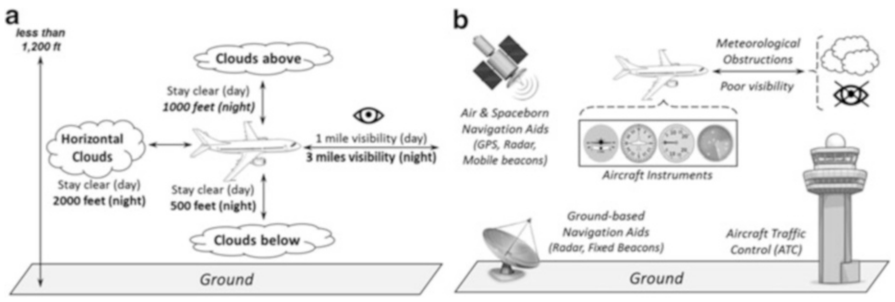

a
less than 1,200 ft
Clouds above
Stay clear (day) 1000 feet (night)
Horizontal Clouds
Stay clear (day) 2000 feet (night)
1 mile visibility (day)
3 miles visibility (night)
Stay clear (day) 500 feet (night)
Clouds below
Ground
b
Air & Spaceborn Navigation Aids (GPS, Radar, Mobile beacons)
Meteorological Obstructions
Poor visibility
Aircraft Instruments
Ground-based Navigation Aids (Radar, Fixed Beacons)
Aircraft Traffic Control (ATC)
Ground

Fig. 56.7 (**a**) shows FAA rules for flight by line of sight (LOS) under 1200 feet altitude (above-ground level), i.e., visual meteorological conditions (VMC) and visual flight rules (VFR) [30, 31]; (**b**) shows instrument meteorological conditions (IMC) and instrument flight rules (IFR) equipment/support infrastructure required when VMC are not met [30]

[6, 29] for collision avoidance is applied, requiring a human operator to maintain situational awareness and deconflict potential collisions. This principle implicitly restricts aircraft operation to manual or semiautonomous control with a human-in-the-loop (i.e., on-board pilot or remote operator).

Practical challenges of remotely piloted aircraft systems (e.g., assurance of continuous remote control) and prospective autonomous aircraft systems (AAS) are driving the modernization of the “see and avoid” principle. The emerging “sense and avoid” principle [6] seeks to address the diminished or complete absence of an HITL. In parallel with this shift, major investments and advancements have been realized toward high-precision and high-reliability sensing technologies to replace and/or supplement human abilities. However, the lack of trusted methodology for machine-based decision analysis and planning drives continued reliance on a HITL. Chiefly, the reconciliation of high safety criticality and a high level of automation (LOA) remains a significant challenge in this application area. Realization of the “sense and avoid” paradigm will require exceeding existing LOA precedents without compromising ATM safety, reliability, or efficiency.

56.5 Future Direction

An agent-based model (ABM) to implement and demonstrate the concept is currently in development. The model is intended to support discrete-event simulation of collision avoidance scenarios for unmanned aircraft systems (UAS) operating in Class G airspace (typically $\leq$1200 ft. aboveground) [31]. The (1) FAA collision avoidance data for man-made static obstacles [32] and (2) geographic information system (GIS) data for natural obstacles (e.g., terrain) and (3) randomized fault injection will be used to emulate airspace environments with uncertain world properties. An autonomous agent representing the system of interest will implement decision analysis and planning logic in response to ABM-generated scenarios. This approach enables characterization of our proposed approach and evaluation against alternative methods. We currently plan to benchmark performance of the proposed approach against leading rule-based and utility-based alternatives for autonomous navigation and aircraft collision avoidance, including: (1) "A-star" (A*)-based pathfinding algorithms [33] such as Dynamic A* Lite (D* Lite) [34] and (2) RTCA DO-185B (traffic collision avoidance system II) – the international standard for aircraft collision avoidance [35].

56.6 Summary

In this paper, we introduced a model for autonomous decision analysis and planning in uncertain worlds (defined as operational contexts with insufficient a priori knowledge available and/or high uncertainty). The concept addresses the challenge of reliable self-management of autonomous systems in response to unanticipated disruptions and/or emergent needs. The model aims to realize a context-adaptation strategy for decision analysis and planning to drive improvised system behavior based on operational context. This strategy contrasts with existing methods that rely on the availability of a priori knowledge and/or exhaustive characterization to optimize system behavior. Lacking this ability in uncertain worlds, our approach aims to drive satisficing behavior [19, 20] and anytime properties [22, 23] in the decision analysis and planning process for enhanced system resilience. Application of the concept to collision avoidance for unmanned and autonomous aircraft is targeted for model verification and validation. This application supports FAA Free Flight initiatives [8, 9] for enhanced safety, reliability, and efficiency in ATM using autonomous systems and processes. Development of an agent-based model (ABM) for implementation, verification, and validation of the concept is forthcoming. Benchmarking against existing methods for autonomous navigation and collision avoidance including A*-based pathfinding algorithms [33, 34] and RTCA DO-185B standard [35] for aircraft collision avoidance is also planned. An interdisciplinary focus with emphasis in systems engineering and decision science is inherent to the overarching scope of research bridging system resilience and system autonomy.

References

1. Madni A, Jackson S (2009) Towards a conceptual framework for resilience engineering. IEEE Systems 3(2):181–191
2. Arsenault D, Sood A (2007) Resilience: a systems design imperative. George Mason University
3. INCOSE (2014) A world in motion - systems engineering vision 2025. San Diego, INCOSE
4. Roberts B, Mazzuchi T, Sarkani S (2016) Engineered resilience for Complex systems as a predictor for cost overruns. Syst Eng 19(2):111–132
5. Cook A, Blom HA, Lillo F, Mantegna RN, Miccichè S, Rivas D, Vasquez R, Zanin M (2015) Applying complexity science to air traffic management. J Air Trans Manag 42:149–158
6. Federal Aviation Administration (FAA) (2016) Integration of civil unmanned aircraft systems (UAS) in the National Airspace System (NAS) roadmap. U.S. Department of Transportation 2012, Washington, D.C.
7. Federal Aviation Administration (FAA) (2016) Aerospace forecast: fiscal years 2016–2036, Washington, D.C.
8. Government Accountability Office (1998) National airspace system: FAA has implemented some free flight initiatives, but challenges remain. (GAO publication no. GAO/RCED-98-246). U.S. Government Printing Office, Washington, D.C.
9. Government Accountability Office (GAO) (2011) National Airspace System: free flight tools show promise, but implementation challenges remain. (GAO publication no. GAO-01-932). U.S. Government Printing Office, Washington, DC
10. Hollnagel E, Woods D (2006) In: Leveson N (ed) Resilience engineering: concepts and precepts. Ashgate, Aldershot
11. Jackson S et al (2012) Resilience principles for engineered systems. Syst Eng 16(2):152–164
12. Lalanda P, McCann J, Diaconescu A (2014) Autonomic computing: principles, design and implementation. Springer, London
13. Krupitzer C, Roth FM, Vansyckel S, Schiele G, Becker C (2015) A survey on engineering approaches for self-adaptive systems. Pervas Mob Comput 17:184–206
14. Kephart J, Chess D (2003) The vision of autonomic computing. Computer 36(1):41–50
15. Arkin RC (2010) The case for ethical autonomy in unmanned systems. J Milit Ethics 9 (4):332–341
16. Arkin RC, Ulam P, Wagner AR (2012) Moral decision making in autonomous systems: enforcement, moral emotions, dignity, trust, and deception. Proc IEEE 100(3):571–589
17. Espinosa H, Ayala-Solares J (2016) The power of natural inspiration in control systems. Springer International
18. Sheridan TB (2011) Adaptive automation, level of automation, allocation authority, supervisory control, and adaptive control: distinctions and modes of adaptation. IEEE Trans Syst Man Cybern A 41(4):662–667
19. Simon HA (1955) A behavioral model of rational choice. Q J Econ 69(1):99–118
20. Simon HA (1956) Rational choice and the structure of the environment. Psychol Rev 63 (2):129–138
21. Sauter VL (2010) Decision support systems for business intelligence, 2nd edn. Wiley, Hoboken
22. Dean TL (1987) Intractability and time-dependent planning. In: Georgeff MP, Lansky AL (eds) Proceedings of the 1986 workshop on reasoning about actions and plans. Calif, San Francisco
23. Dean TL, Boddy M (1988) An analysis of time-dependent planning. In: Proceedings of the seventh National conference on artificial intelligence. Calif: American Association for Artificial Intelligence, Menlo Park, pp 49–54
24. Russell SJ, Zilberstein S (1991) Composing real-time systems. In: Proceedings of the twelfth international joint conference on artificial intelligence. Calif: International Joint Conferences on Artificial Intelligence, Menlo Park, pp 212–217

25. Norbis MI, Smith JM (1988) A multiobjective, multi-level heuristic for dynamic resource constrained scheduling problems. Eur J Oper Res 33(1):30–41
26. Glover F (1989) Tabu search—part I. ORSA J Comput 1(3):190–206
27. Glover F (1990) Tabu search—part II. ORSA J Comput 2(1):4–32
28. International Business Machines (IBM). An Architectural Blueprint for Autonomic Computing (2005) Hawthorne. IBM Corp, New York
29. Speijker L, Verstraeten J, Kranenburg C (2012) Scoping Improvements to 'See And Avoid' for General Aviation (SISA) (Rep. No. NLR-CR-2012-362). European Aviation Safety Agency (EASA)
30. Basic Visual Flight Rules (VFR) Weather Minimums (2004) 14 C.F.R Section 91.155
31. Federal Aviation Administration (FAA) (2016) FAA-H-8083-258, pilots handbook of aeronautical knowledge. United States Department of Transportation, Oklahoma
32. Federal Aviation Administration (FAA) (2016) Digital Obstacle File (DOF), June 19, 2016 [Data set]
33. Hart P, Nilsson N, Raphael B (1968) A formal basis for the heuristic determination of minimum cost paths. IEEE Trans Syst Sci Cybern 4(2):100–107
34. Koenig S, Likhachev M (2005) Fast replanning for navigation in unknown terrain. Trans Robot 21(3):354–363
35. Radio Technical Commission for Aeronautics [RTCA] (2008) Minimum operational performance standards for traffic alert and collision avoidance systems II (TCAS II) (DO-185B). RTCA, Washington, D.C.

Chapter 57
Model-Centric Decision-Making: Exploring Decision-Maker Trust and Perception of Models

E. Shane German and Donna H. Rhodes

Abstract Ongoing research is exploring various dimensions of enabling model-informed decisions, as motivated by the increasing need for individuals and teams to make decisions based on models and model-generated information. Central to this topic is the need to understand what engenders trust in models. This exploratory study uses expert interviews to investigate how various types of decision-makers and actors interact with and use models, including to what degree models are used to inform system decisions and how individuals build trust in models. While anecdotal stories of success and failure exist, empirical studies are needed to truly understand the many facets of human decision-making in model-centric engineering. Such research is expected to generate key insights that can inform current and future practice, as well as determine areas for more extensive study.

Keywords Interactive Model-Centric systems engineering • Model-centric • Decision-making • Sociotechnical • Trust • Interviews • Transparency

57.1 Introduction

Models are increasingly used to drive major acquisition and design decisions, yet model developers, analysts, architects, program managers, and senior decision-makers are faced with many challenges. Blackburn et al. captured many of these challenges in an investigation of the technical feasibility of radically transforming systems engineering through model-centric engineering [1, 2]. Digitized legacy systems and new digital system models will provide the basis for designing and evolving systems in the future [3]. This drives the criticality of models as assets and necessitates change in model-related policy and practices [4]. The Model-Centric Engineering Forum conducted by the U.S. Department of Defense (DoD) Systems Engineering Research Center (SERC) in May 2016 fostered a dialogue between

E. Shane German • D.H. Rhodes (✉)
Massachusetts Institute of Technology, Cambridge, MA, USA
e-mail: esgerman@mit.edu; rhodes@mit.edu

A.M. Madni et al. (eds.), *Disciplinary Convergence in Systems Engineering Research*, https://doi.org/10.1007/978-3-319-62217-0_57

industry, government, and academia on current state of practice and vision for transformation [5].

The Interactive Model-Centric Systems Engineering (IMCSE) research program, initiated in 2014, aims to inform and contribute methods, processes, and tools to improve human–model interaction, in support of accelerating the transition of systems engineering to a more model-centric discipline [6]. IMCSE advances knowledge relevant to human interaction with models and model-generated information, drawing from relevant knowledge from other fields (e.g., cognitive science, visual analytics, data science), placing it within the context of systems engineering. Additionally, this research generates knowledge impacting human effectiveness in model-centric environments of the future [7]. As part of this exploration into human–model interaction, German and Rhodes examined the transition from traditional aircraft cockpits to modern glass cockpits as an analogy case, indicating information abstraction and automation led to new cognitive and perceptual challenges. Non-technical factors will have significant influence in the future of digital system models, including trust, buy-in, and belief [8].

57.1.1 Motivation

The research discussed in this paper explores various dimensions of enabling model-informed decisions, as motivated by the increasing need for individuals and teams to make decisions based on models and model-generated information. Models represent an abstraction of reality in order to make predictions about the future, based on assumptions. Models can come in a variety of forms and formats, but fundamentally are an encapsulation of reality that humans use to augment their ability to make sense of the world, anticipate future outcomes, and make decisions. Among the many challenges are reasoning, comprehension, and collaborative decision-making in the face of uncertainty, combining artificial (model-generated), and real data, and effectively utilizing vast amounts of information.

Significant progress continues to be made in the theory and practice of model-based engineering, yet little attention has been given to the complexities of human–model interaction. An open area of inquiry relates to the various facets of humans interacting with models and model-generated information throughout the lifecycle. The 2015 IMCSE Pathfinder Workshop validated the belief that improving human–model interaction would significantly improve model-centric engineering [9]. Additionally, a 2016 workshop report sponsored by the National Science Foundation (NSF), the National Aeronautics and Space Administration (NASA), the Air Force Office of Scientific Research (AFOSR), and the National Modeling and Simulation Coalition (NMSC) highlights the need for understanding the individuals involved in the modeling process and how these individuals affect model development and usage [10].Central to this is the need to understand what engenders trust in models. While anecdotal stories of success and failure exist, empirical studies are needed to truly understand the many facets of human decision-making in model-centric engineering.

57.2 Research Approach

This study aims to generate insight into decision-maker trust and perception of models and model-generated information through expert interviews. Experts in system decision-making accumulate various kinds of knowledge and wisdom, often through years of hard-earned experience. Rather than theorize on how various actors interact with and trust models, this interview-based approach allows us to gather qualitative, empirical data by asking them directly. This study is ongoing and is not meant to offer definitive truth for all types of decision-making with models, but rather to serve as an exploratory study into a little-researched area. This study is primarily scoped to decision-makers and systems experts found within the defense and aerospace communities.

57.2.1 Sampling

Unlike quantitative research, which advocates random sampling approaches, qualitative research seeks to "select 'information-rich' respondents who will provide you with the information you need" [11]. For this study, we have primarily used judgmental and expert sampling to identify "persons with demonstrated or known expertise in an area of interest," [11] along with individuals who, although perhaps not widely known as "experts," were judged to have experience relevant for achieving the objectives of the study. In this study, we broadly view an expert as an individual who works or has worked as an actor within model-based decision-making processes, and can provide knowledgeable insight and perspective informed through his or her experiences. The definition of an expert is clearly open to interpretation as an "expert" may very well be in the eye of the beholder, and an improper interpretation on the part of researchers may lead to a biased sample of participants that fails to adequately represent a population. From our perspective, however, all participants were judged to have relevant experience and credentials through either their individually known work with models or that of the organizations for which they have worked, primarily experience found within the domains of defense and aerospace. While the study is ongoing, 30 individuals have been interviewed at the time of writing.

Interviews for this study followed a semi-structured format that allowed interview participants latitude to share a wide range of perspectives and insights while following guiding questions aimed at generating insight for the study objectives. Table 57.1 presents a list of the general questions asked.

Table 57.1 List of interview questions

1. What types of decisions do you make, or help others make, with models?
2. What is the degree to which the decisions you make are based on models?
3. Do you view models as a primary or supplementary source in decision-making?
4. How do you develop trust in models?
5. How do you judge if a model can be trusted?
6. How much transparency do you desire?
7. What factors have led to inappropriate trust in models?
8. What limits your ability to use models to make decisions?
9. What challenges or failures have you experienced with the use of models in system decision-making?
10. What approaches or policies have been applied, or would you like to see applied, to mitigate those challenges?
11. How desirable would the ability be to directly interact with models real-time while making decisions?

57.3 Decision-Making Flow of Model-Generated Information

High-level decisions incorporating an explicit model in the decision-making process include the following broad components:

1. A model that represents some aspect of the system of interest
2. Human actors
3. A decision to be made

While simplified, a generic conceptualization of the model-influenced decision-making process is helpful for facilitating discussion surrounding this research space. In this general framework, the information generated from a model is the common thread that connects the three generic commonalities listed above. First, a model must be created or already exist before it can be of use in a decision-making context—this creation of the model itself generates information relevant to decision-making before it is even "used." Next, the model in question generates information designed to facilitate a better understanding of an issue for which a decision must be made. Exactly what happens to this model information varies from context to context, but all contexts involve model information flowing *from* an actor (i.e., modelers or analysts directly interacting with the model), *through* another actor or actors, and ultimately *to* a final decision-making actor. Where decision-makers reside in the process seems to be more along a spectrum of the flow of model information. Within different decision-making contexts, actors may even find themselves in different roles. For example, in a mid-level decision-making context, an actor may be the individual *to* whom the information is flowing, yet in a higher level decision, the same individual may become a *through* actor. In decisions involving more than one actor, however, all model-informed decisions involve information being generated and flowing from those directly interacting with a

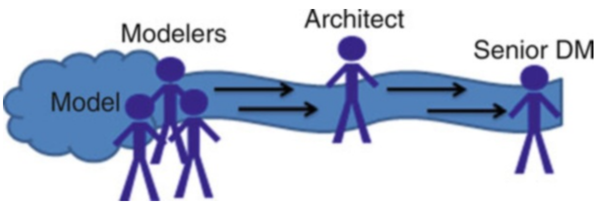


Fig. 57.1 Example flow of model-generated information

model, flowing through an actor or actors, and lastly reaching a final decision-maker to whom the information flows.

To elucidate this conceptualization, it may be helpful to examine a specific case example that illustrates this flow of information. Figure 57.1 illustrates one such scenario where a model is used to inform decision-makers in a war game. The senior level decision-maker (Senior DM) identifies a modeling need, and interfaces with a model architect to create the desired model. The architect works with a team of modelers who develop and test the model and produce model outputs that are communicated through the architect to the decision-maker in response to specific queries. In this case, the model information flows *from* the team of modelers who comprise the initial actors, then flows *through* the primary model architect, and finally *to* the Senior DM involved with the war game.

At the end of the model-generated information flow, there is a fairly discrete decision or set of decisions to be made. These high-level decisions, however, are influenced by countless smaller decisions and actions performed by various individuals within the flow of information. This study seeks to better understand the perspectives and thought processes of these various actors with the hope of better understanding the decision-making process as a whole. In the sampling process, we sought perspectives from individuals from all three of our conceptualized categories; however, for the purposes of this paper, *through* individuals comprise the majority of participants. While this study is ongoing, we believe the 30 individuals interviewed at the time of writing present enough information to warrant publishing of the current results.

57.4 Trust

Ricci et al. describe how trust in models relates to a user's perception of how close to a specified reality a constructed model is perceived to be. Ultimately, a good decision "is one based on a trusted, truthful representation of both reality and values" [12]. The 2015 IMCSE Pathfinder Workshop report notes that numerous challenges exist within model-centric development, including challenges surrounding "perception of truthfulness and trust" in models, as this aspect of trust can ultimately affect "the timeliness, quality, and confidence in model-based decisions" [9]. The Pathfinder report also expresses a desire not just for models to be trusted, but for that trust to be supported with underlying evidence [9]. Blackburn et al.

articulates a vision for developing model-centric environments into a “single source of technical truth” for decision-makers [2]. West and Pyster communicate the idea of digital system models offering an “authoritative representation” of systems [3]. Gass and Joel note, however, that all models “reflect modelers’ views of how the decision problem can be resolved,” and these views carry inherent assumptions and limitations that decision-makers must consider prior to determining if the subsequent modeling results appropriately align with their decision at hand [13]. With this in mind, the goals of developing single sources of “truth” and “authoritative data” will require decision-makers to evaluate and determine how much trust they should place in this data. This trust can be improperly calibrated, however, potentially resulting in overreliance or underutilization. Engendering an appropriate level of trust within decision-makers is crucial to effective use of models in decision-making.

Literature addressing human trust in automation offers insight that can be useful when applied to this discussion on human trust in models. This relationship seems rather natural when considering that automation may arguably be nothing more than a model of operation algorithmically programmed into a machine. In the article “Humans and Automation: Use, Misuse, Disuse, Abuse,” Parasuraman and Riley highlight multiple potential pitfalls to consider when placing humans into interaction with automation. Misuse is defined “as overreliance on automation (e.g., using it when it should not be used, failing to monitor it effectively), disuse as underutilization of automation, [. . .] and abuse as inappropriate application of automation by designers or managers” [14]. While examining factors that may contribute toward use and application of automation, Parasuraman and Riley identify that “trust often determines automation usage” [14]. This taxonomy of use, misuse, disuse, and abuse can provide a useful framework for thinking about how humans interact with complex models as well.

But what exactly is meant by “trust?” Lee and See define trust as “the attitude that an agent will help achieve an individual’s goals in a situation characterized by uncertainty and vulnerability” [15]. Specifically addressing misuse and disuse, Lee and See express that “[o]vertrust is poor calibration in which trust exceeds system capabilities; with distrust, trust falls short of the automation’s capabilities” [15]. This idea of calibration “refers to the correspondence between a person’s trust in the automation and the automation’s capabilities.” Trust in automation implies belief that the automation will do what it is supposed to do, while trust in models assumes that the models will provide the information you want. Both automation and models represent technologies that require a certain amount of trust as the underlying processes and assumptions may be difficult to fully understand. The goal is not just for models to be used but to be used appropriately; models, much like automation, have limitations of effectiveness and applicability. Overreliance in models can lead to misuse by inappropriately applying models outside of their inherent limitations. Conversely, improper lack of trust in models can lead to decision-makers discounting relevant model information that could have otherwise aided in the understanding and solution of issues. By examining the human aspect of human–model interaction, this study aims to generate understanding that can lead to appropriate "calibration" of human trust in models. Before seeking to influence the human actors, however, it is necessary to understand how those actors actually work in practice.

57.4.1 Developing Trust

Consciously or not, decision-makers must have a certain amount of trust present before model-generated information is used in the decision-making process. Few of the actors interviewed have consistent processes to develop trust in new models, yet all have various factors they consider when determining trust. Some factors prove unique to specific individuals or groups of individuals along the flow of information, while other factors appear to be common for individuals throughout the entire flow. In addition to processes or factors influencing decision-maker trust, we want to know more about what specific attributes or types of information about models that decision-makers and actors care about knowing.

57.5 Key Findings

This section presents preliminary key findings of the study to date. While these findings may not necessarily be novel, they serve to form a compilation of empirical evidence concerning human–model interaction. As this work is ongoing, these findings are expected to grow and evolve. The results are presented in no particular order of importance.

57.5.1 Technical and Social Factors Influencing Trust

As summarized by one participant, "trust is terribly important" within the modeling and decision-making process. While few of the interviewed experts have a specific process used in determining trust, every participant has various factors that they consider while determining the amount of trust to put in a model. This trust is also very contextually dependent, meaning that the trust is not so much in the model as an entity, but in the usefulness of the model for a specific decision at hand. Various factors influence individuals' trust in models, yet these factors may vary in importance depending on the specific individual involved. A clear theme that has emerged from the interviews, however, is that both technological and social factors come into play when determining the amount of trust that any type of actor is willing to place in a model. In many cases, the importance of technological factors appears to diminish in relation to social factors as actors move further along the flow of model information. Figure 57.2 illustrates the concept that various

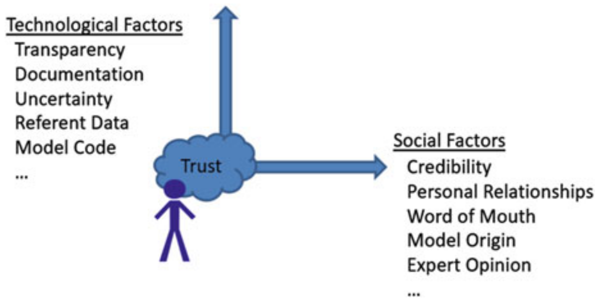


Fig. 57.2 Sociotechnical factors influencing trust

technological and social factors influence a decision-maker’s trust in a model. The factors listed are not all-inclusive, but represent some of the factors identified through the interviews. While there may be trends in comparing important factors between the “*from, through, to*” categorizations of actors, such as the generalization that social factors seem more salient for *to* actors than for *from* actors, this is still dependent on the specific individuals involved. A strongly supported generalization, however, is that both technological and social factors play an important role in influencing an individual’s trust, and any attempt to understand trust without considering both types of factors would be lacking.

57.5.2 Importance of Communication

Communication arose as a key attribute of effective model decision-making. Before any effective modeling can be accomplished, senior-level decision-makers must construct the problem statement clearly and in a form that unambiguously expresses the information they desire from models. Oftentimes the problem can change, however, therefore consistent communication of the problem at hand is crucial for allowing individuals below them to create or use models to generate relevant and useful information. The onus for this specific communication does not fall solely on senior levels, however, and lower levels must actively update senior decision-makers on progress to gain feedback on whether they are addressing the actual problem. Senior decision-makers must likewise be open and available, to the extent possible, to provide this feedback as necessary. As noted by an interview subject, “models [. . .] bring their own language with them” that can create communication barriers that stifle decision-makers’ understanding of the model output presented to them. Unless a decision-maker is similarly an expert in the model, there needs to be a “translation between output to decision-maker speech,” before the information can usefully be incorporated by the final decision-maker. Modeling aims to provide an asset for a decision; however, this asset cannot be effective if it is not useful for a decision-maker, and it cannot be useful if not understood. Instead of relegating discussion between actors to the beginning and end of a decision-making process, employing continuous and iterative communication may further reduce the acceptance barrier by allowing decision-makers to feel as if they walked the up-stream actors to the final model outputs. The flow of information between actors, including both expression and interpretation of information, must be intentional and unambiguous.

57.5.3 Transparency

Most of the interviewed modelers, analysts, and architects emphasized the importance of having access to precise technical information of models, oftentimes stating a desire to have access to code and the "guts" of the model in question. One such practitioner expressed that he "hopes everyone wants full transparency," seeming to assume that the desire for full transparency is a given for anyone making decisions from models. Transparency serves to enable an understanding of how a model actually works in order to determine if the model should be used for a specific decision. The understanding of a model encompasses, but is not limited to, a model's code, and transparency should include access into practices and decisions involved in creating and validating the model. Moving further along the flow of information decision-making, however, precise information about the models may become less desired, and even unwanted. Comments such as "I trust the people below me" convey the paradigmatic shift that occurs. While details such as model assumptions and uncertainties remain desired, the need for intimate technical knowledge seems to fade. Responses suggest that, even if an actor does not personally require full transparency into a model, transparency should still be available to trusted actors before them in the flow. This suggests a significant point: as actors move further along the flow of information and have less time and ability to personally investigate a model and build their own trust in the model, their trust instead shifts more onto their people to investigate the model for them. In this understanding, the trust for decision-makers is "implicitly on the models, but explicitly on the people."

57.5.4 Understanding of Assumptions and Uncertainty

All models are inherently abstractions of reality that contain assumptions and uncertainties. Models are created for a specific reason and context, and while the assumptions within the model aim to help answer those questions, they also fundamentally create bounds of model applicability. Failure to properly understand the inherent limitations found within a model increases the likelihood of the model being used inappropriately. "All models are wrong, but some are useful," [16] and before any can be useful, their limitations must be understood. As models cannot perfectly encapsulate and relate the situation of interest, uncertainty is fundamentally a part of the results, and uncertainty is also fundamentally a part of determining if the results are appropriately relevant to the decision to be made. This uncertainty must be sought after, understood by the sources of model information, and then passed clearly along the flow of information. There is a fundamental need to understand and express model uncertainties throughout the decision-making flow. Organizational and social dynamics can hinder this expression of uncertainty, however. In some instances, uncertainty about an answer may entail negative stigmas and imply failure to do one's job correctly. Decision-making cultures need to strive to drive out fear of uncertainty expression and transparency. The tragedy of the space shuttle Columbia offers a painful reminder of what can happen when important information is not effectively passed along the decision-making flow [17].

57.5.5 Documentation

Model developers internally carry within themselves the most intimate knowledge of a model's limitations and capabilities. Similar to how modeling is a process of making the internal mental models and expertise found within individuals explicit, documentation is a process of making the assumptions and limitations of a model explicit. Models may very well be validated, even accredited; however, this validation and accreditation are for specific conditions, outside of which the model is no longer valid. Multiple interviews revealed the danger of assuming a model can extend to any context needed when in fact its appropriate contexts of use are much more limited. For a model to have any sort of reuse capability, these assumptions and limitations should be documented in an accessible way so that others can understand how they might appropriately apply the model to their specific situation. Models are built to answer a specific question or set of questions, and the early conceptualizations (e.g., whiteboard drawings) of the model and decisions made in the development process can provide important insight into understanding the model in addition to the documentation of assumptions within the model itself. These conceptualizations, if captured, can provide useful artifacts in the understanding and trust of a model. As models become more complex, documentation of assumptions and capturing of conceptual artifacts and decisions will likely prove crucial in allowing actors to appropriately calibrate their own understandings and mental models of if, and how, a model should be applied to specific decision-making scenarios.

57.5.6 Primary Versus Supplementary

Of the experts interviewed, distinctions emerge concerning the primacy of explicit models in the decision-making process. Some view models as clear primary sources in decision-making, others adamantly express that they should only be supplemental sources, and still others present the oft-favored viewpoint of systems engineers—it depends. Those that favor models as a primary source in decision-making point to the benefits of increased knowledge and insight that models can provide if done correctly. Others that advocate for supplementary use emphasize the danger of abdicating the decision-making process to models, and point to the inability of models to capture every relevant factor in a decision. One participant noted an increasing reliance on modeling and simulation (M&S) in decision-making, unfortunately accompanied with the increasing desire to rely on M&S without having to "understand the fundamental processes behind it." The variations in responses serve to validate the non-definitive (yet still insightful) answer of "it depends." Truly, how models are viewed and used is dependent upon the model users and decision-makers, along with the modeling and decision-making context. Well-established and validated physics-based models, for instance, might prove to be a primary source in a decision-making scenario, while descriptive or predictive models that are less conducive to traditional validation may contribute more of a supplemental input within a wide range of other inputs.

57.5.7 Independent Review

Although models strive to reduce complexity of reality to understandable and workable abstractions, they can still be very complex. Verification ("Did I build the thing right?") and validation ("Did I build the right thing?") (V&V) are crucial for determining the efficacy and relevancy of a model for decisions [18]. Just as skill is needed in model development and use, checks like V&V are required to hopefully catch the inevitable errors. However, effective V&V likewise requires skill and is liable to its own errors. One longtime system architect we interviewed emphasized the importance of utilizing independent experts who can review and render judgments concerning the credibility of results and believability of the data used. Such a team would be composed of individuals with areas of expertise relevant to the problem. One might view the team as analogous to a forensics team that closely examines the data and code being used and makes judgments that assess the efficacy of decisions made along the flow of model information. Depending on the model and decision-making context, the format and formality of reviews could range from formal, externally based reviews, to informal, internal peer reviews within a team. Whatever the format, a form of review can serve an important part in the creation of an effective model and as such should be a process that is transparent to the decision-makers who are ultimately affected.

57.5.8 Investment Bias and Politics

One individual related the story of a program that involved significant investment in modeling and simulation. When the time came for program decision-makers to make a decision, “they had no choice but to accept” the model’s answers “given the resources that were spent.” Such a story brings to light the potential bias that investment of time and resources into model development will yield correct and reliable results. Further interviews also revealed a potential for decision-makers to use money as a basis for establishing trust in model results. Money may sometimes offer a useful indicator of model capability; however, no matter how much money is spent on a model, the model is still bounded by the problem space it was designed to solve. Just because large amounts of money were spent on a model does not mean that it is appropriate for the decision at hand. If this issue is not a bias in some cases, then perhaps it may be a political pressure to make a decision based off the model results because of the money spent on model development—if not, the money was wasted. Such a logical fallacy should be countered by a fundamental term of economics: sunk cost. Once money and resources have been spent (sunk) they are gone, and no longer should have any bearing on decisions seeking to promote benefit in the future.

57.5.9 Confirmation Bias

In the words of one respondent: “Quite often, what I see is that decision-makers use models as confirmation bias.” This statement reflects one potential pathway for models to be used inappropriately, namely, as a means to further one’s own preconceptions or agendas that may be incorrect. Just because a decision-maker’s intuition for a solution matches up with a subsequent modeling result does not mean the intuition or the modeling was wrong; in fact, it could be a testament to the decision-maker’s experience. However, a senior modeler noted the challenge of guarding against bending a model and results to produce answers desired by decision-makers. Another participant expressed the “amusing thing” that in high-level war game simulations, the war games “almost always” are eventually modified so that your side wins. These interviews reflect the importance for all actors to honestly seek truth while participating in the modeling process. Modeling aims to provide solutions to problems; however, if generated and used to advance one’s agenda or to inappropriately confirm preconceived notions, the “solutions” provided may in fact be more damaging than if models were not used in the first place.

57.5.10 Endogeneity of the Human

Underpinning this study has been the clear and consistent theme expressing the endogeneity of the human in the model-centric decision-making process. Many senior decision-makers do not have the bandwidth, training, or time to become technical experts in the models that are used to inform their decisions. How do they trust complicated models? As one senior-level decision-maker put it: "The answer is they trust the people." They trust that the people before them in the model information flow handled the data correctly, created, tested, used, and analyzed the model correctly, and expressed the results accurately with appropriate information on uncertainties, assumptions, and limitations. Decision-makers trust that those individuals have the appropriate expertise and capability to understand and address the problem at hand. On the other hand, senior decision-makers also need to have the technical judgment to be able to "sniff out" the wrong answers, and have a healthy technical competence appropriate to the decisions being made. As systems and their models become more and more complex, the need for skilled and experienced individuals to work within the flow of information seems to be more necessary than ever. Yet, the inevitability of aging and retirement guarantees that the experts of today cannot be the experts of tomorrow. Without the right people capable of handling the complexities we are creating, the system will fail, regardless of the technology and innovation we throw at it.

57.5.11 Real-Time Interaction with Models

A final question we asked in the interviews concerned the desirability of being able to directly interact with models in real time while making decisions. Overwhelming, the respondents view interactivity with models as highly desirable. After all, many decisions involve asking "what-if" questions about the model, and direct interaction could serve to gain insight, build intuition, and speed innovation without needing to go through other human actors. However, This support for model interactivity also comes tempered with caution from some individuals. Specifically, caution against allowing actors interactive access to models without a calibrated understanding of the model's capabilities and limitations. As related by one individual, in situations without this appropriate understanding, "I can get lots of results real quick, and I can make lots of bad decisions real fast." These interviews make abundantly clear the importance for properly understanding a model and its associated assumptions before determining one's trust and usage of model results. Such an understanding is crucial for effective and appropriate interaction with models. As stated in another interview, "If you make it so fools can use it, fools will use it." So while direct interaction with models may be rightly desired based off its potential benefits, development and deployment of interactive models must also advance in a smart and conscientious manner to ensure that actors are not being set up for failure due to ignorance of their own limitations.

57.6 Discussion

The increase we see in system modeling is driven both by a desire to better understand complex systems and issues as well as by increases in technological and computational capability. Similar to technical modeling in many ways, automation involves increasing automation in systems as advancements in technology allows. Often, this increase results in gains of efficiency and safety, yet the history of automation has also shown that humans are not just outside users of systems, but rather are endogenously critical components of the system. Experience has also shown that increasing technological capability for the sake of technical achievement, without proper consideration for the human component, can have dire consequences. Bainbridge writes about the "ironies of automation," where introducing automation can sometimes increase the workload and complexity of tasks it aimed to reduce [19]. With gains in modeling complexity and capability pointing to a model-centric paradigm of engineering, we should be cognizant of potential "ironies of modeling" where failure to appropriately account for human decision-makers and actors results in worsening decision-making processes we aimed to improve.

57.7 Future Research

This study aims to generate empirical insight into how human actors interact with and trust models, while also providing a starting point for continued exploration into how human actors and decision-makers trust, perceive, and interact with models. Through the interviews conducted, we hope to identify important considerations surrounding human–model interaction and trust that experts deem important for effective model use and decision-making. These considerations include practices that interviewed experts implement to aid in their decision-making, along with identified challenges and potential mitigations to challenges that can degrade effective model-centric decision-making. The insights gained from these interviews are planned to be coupled with empirical case studies examining human interaction with complex, abstracted systems to gather information about how human actors and decision-makers actually perform in practice. The descriptive insights gained through empirical research will be bolstered with normative research on decision-making and biases. Taken together, we envision these various threads of research weaving together toward prescriptive outcomes of heuristics and design principles to inform policy, design, implementation, and use of model-centric engineering.

Acknowledgments This material is based upon work supported, in whole or in part, by the U.S. Department of Defense through the Systems Engineering Research Center (SERC) under Contract HQ0034-13-D-0004. SERC is a federally funded University Affiliated Research Center managed by Stevens Institute of Technology. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the United States Department of Defense.

References

1. Blackburn M, Cloutier R, Witus G, Hole E, Bone M (2015) Transforming system engineering through model-centric engineering. Stevens Institute of Technology, SERC-2015-TR-044-3
2. Blackburn M, Bone M, Witus G (2015) Transforming system engineering through model-centric engineering. Stevens Institute of Technology, SERC-2015-TR-109
3. West TD, Pyster A (2015) Untangling the digital thread: the challenge and promise of model-based engineering in defense acquisition. Insight 18:45–55
4. Zimmerman P (2015) MBSE in the Department of Defense. Office of the Deputy Assistant Secretary of Defense for Systems Engineering, US Department of Defense
5. Clifford MM, Blackburn M, Verma D, Zimmerman P (2016) Model-centric engineering – insights and challenges: primary takeaways from a government-industry forum. Stevens Institute of Technology
6. Rhodes DH, Adam MR (2015) "Interactive Model-Centric Systems Engineering (IMCSE) Phase 3." Stevens Institute of Technology, SERC-2015-TR-043-1
7. Rhodes DH, Adam MR (2016) A vision for human-model interaction in interactive model-centric systems engineering. Presented at the INCOSE international symposium 2016, Edinburgh
8. German, ES, Donna HR (2016) Human-model interactivity: what can be learned from the experience of pilots with the glass cockpit? 14th conference on systems engineering research, Huntsville
9. Rhodes DH, Adam MR (2015) Interactive model-centric systems engineering. Pathfinder Workshop Report
10. Research challenges in modeling & simulation for engineering complex systems. National Science Foundation Workshop Report, January 13–14, 2016
11. Kumar R (2011) Research methodology: a step-by-step guide for beginners, 3rd edn. SAGE, London. Print. 2011
12. Ricci N, Michael AS, Adam MR, Donna HR, Matthew EF (2014) Exploring stakeholder value models via interactive visualization. 12th Conference on Systems Engineering Research, Redondo Beach
13. Gass SI, Lambert SJ (1981) Concepts of model confidence. Comput Oper Res 8(4):341–346
14. Parasuraman R, Riley V (1997) Humans and automation: use, misuse, disuse, abuse. Hum Factors 39(2):230–253
15. Lee JD, Katrina AS (2004) Trust in automation: designing for appropriate reliance. Humans Factors 46(1):50–80
16. Box GEP, Norman RD (1987) Empirical model-building and response surfaces. John Wiley & Sons, New York
17. Smith MS (2003) NASA's space shuttle Columbia: synopsis of the report of the Columbia accident investigation board. Congressional Research Service, the Library of Congress, Washington, DC
18. Pace DK (2004) Modeling and simulation verification and validation challenges. J Hopkins APL Tech Dig 25(2):163
19. Bainbridge L (1983) Ironies of automation. Automatica 19(6):775–779

Chapter 58
Implementing Value-Driven Design in Modelica for a Racing Solar Boat

Joshua Sutherland, Alejandro Salado, Kazuya Oizumi, and Kazuhiro Aoyama

Abstract Research has shown that current design approaches, such as requirement-based design or cost as independent variable (CAIV), may fundamentally yield suboptimal designs. In response to the need for better systems, new design techniques that are based on optimization and decision-making have been proposed. In this paper, we show how Modelica can be used to implement and operationalize value-driven design (VDD) in concept selection. Modelica's object-oriented strengths are employed to model design alternatives and its capability to execute Monte Carlo simulation enables the introduction of uncertainty in models and assessment. The proposed approach has been applied to the conceptual design of an unmanned, autonomous solar powered boat, which is aimed at racing in a student competition. Value has been defined as a function of the probability to win the said race, which expands usual examples of value functions to nonmonetary ones. This paper describes the approach as well as the benefits, limitations, and obstacles encountered during its implementation.

Keywords Value-driven design • Conceptual design • Modelica • System modeling • Monte Carlo

J. Sutherland (✉) • K. Oizumi • K. Aoyama
Department of Systems Innovation, Graduate School of Engineering, University of Tokyo, Tokyo, Japan
e-mail: joshua@m.sys.t.u-tokyo.ac.jp; oizumi@m.sys.t.u-tokyo.ac.jp; aoyama@m.sys.t.u-tokyo.ac.jp

A. Salado
Grado Department of Industrial and Systems Engineering, Virginia Tech, Blacksburg, VA, USA
e-mail: asalado@vt.edu

A.M. Madni et al. (eds.), *Disciplinary Convergence in Systems Engineering Research*, https://doi.org/10.1007/978-3-319-62217-0_58

58.1 Introduction

In system development, early stage conceptual design plays a significant role in the chances to succeed in maximizing the value associated to the users' perceived satisfaction [1]. Research has shown that current design approaches, such as requirement-based design or cost as independent variable (CAIV), may fundamentally yield suboptimal systems [2]. In response to the need for better systems, the value-driven design (VDD), which is built on the pillars of optimization and decision-making, has been proposed [2, 3]. In VDD, a value function that relates system attributes to the level of satisfaction that is experienced by the corporation developing the system, usually through a user's demand function, is defined. Then, a particular design is optimized by modifying system attributes and evaluating their effect on system value [2].

This paper builds upon the authors' previous work on creating and assessing conceptual designs by means of hierarchical functional decomposition in object process methodology [4, 5] and subsequent assessment by way of the simulation of Modelica models [4–6]. It provides two primary contributions. First, it integrates the VDD methodology explicitly within Modelica's object-oriented design environment. And second, while most of published applications of VDD have employed financial value functions, primarily economic profit, the work presented in this paper utilizes a nonmonetary value function (the probability of winning the race), which helps in generalizing the application of VDD. This is applied to the early stage design of an unmanned solar racing boat for a student competition as a test case.

58.2 Background

58.2.1 *Value-Driven Design (VDD)*

In VDD, a particular design is optimized by modifying system attributes and evaluating their effect on system value [2]. Value is defined in this framework as a level of satisfaction that is experienced by the corporation developing the system and is usually monetized to profit. In decision-based engineering design, the attributes of a system are modified until the utility of the system is maximized, with respect to corporate preferences [7]. In this case, the definition of utility is directly taken from economics [8].

This contrasts with requirements-based design where prior to completing the design it is decided what performance the system and its components must exhibit and thus ignoring the uncertainty fundamentally associated with creating a new system [2]. Given the magnitude of the change required in systems engineering to move from requirements-based design to VDD, a workshop was held by the US National Science Foundation in 2010 to compare the two approaches, the content of which is described by Collopy et al.[9] It is suggested that while requirements-based approaches are embedded in industry today successfully, allowing for contracting between customers and suppliers, they continue to fail at capturing what the customer prefers and prevent the search for better designs. While the concept of VDD is embraced by academia, it remains a challenge to operationalize it in real projects. Hence, further research is required.

58.2.2 System Modeling

A large amount of engineering effort is spent in developing models of designs such that prediction can be made of their expected performance before they are built, avoiding costly trial and error. SEBoK [10] lists various types of models divided between abstract models and physical models, with abstract models being further divided into descriptive models (describing logical relationships) and analytical models (describing mathematical relationships). Dynamic models are a subcategory of analytical models and are appropriate for modeling the performance of systems with time-varying states and as such clearly applicable for modeling vehicles in motion.

The ease at which sufficiently accurate models can be created is clearly an important consideration on any project and object-oriented approaches have successfully enabled the software industry to develop ever more complicated products by building and utilizing libraries enabled by object-oriented technology. For engineers looking to create dynamic models in an object-oriented paradigm, Simulink and Modelica are both viable options and both popular in industry for the design of vehicle systems.

In Modelica and Simulink, the behavior of components is captured in equations and these components are connected together to develop subsystems and ultimately form the system being designed. While superficially sounding similar, Modelica offers benefits over Simulink for this research. Primarily, as components can be connected with acausal connections (e.g., electrical current can flow in both directions) and physical equations can be simply declared with the solver handling how to execute them, Modelica enables the engineer to focus on the physical reality of the systems being designed. This avoids some errors associated with attempting to code a model of the system.

58.2.3 Modelica and VDD

Modelica has been used extensively in industry and research to assess the performance of systems before they are realized physically. Reviewing the most cited past work, it tends to be focused on describing the language (e.g., [11, 12]), developing highly accurate modeling libraries for a particular domain (e.g., [13, 14].) or on the generation of Modelica models from other modeling languages (e.g., [15]). Modelica has been used as the assessment mechanism for design methodologies in the past such as described by Starling and SheaK [16], but such approaches focused on multiobjective optimization as opposed to supporting VDD.

Further literature review found one example attempting to describe VDD assessment being performed using Modelica by Du et al. [17]. However, Du's work does not consolidate the value quantities and instead plots performance of various designs against surrogates of maintenance cost and capital cost. Hence, there is no attempt to rank the designs and select the one that provides the most value.

58.3 Methodology

An overview of the approach advocated in this paper is shown in Fig. 58.1, which for illustration shows the solar boat case study (referred to in this paper as SolarBoat).

First, we define the value function of the system being designed, which for SolarBoat we declare as the probability of winning the race subject to costing not more than a fixed budget.

On deciding the external factors interacting with all SolarBoats, it is possible for the value function and the infrastructure to assess each SolarBoat design modeled in

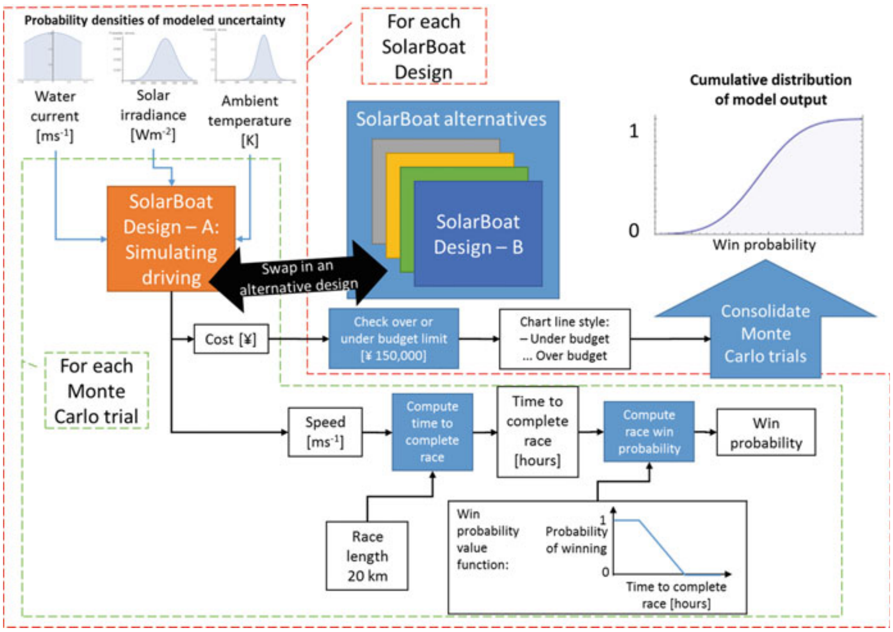

Probability densities of modeled uncertainty
Water current [ms^{-1}]
Solar irradiance [Wm^{-2}]
Ambient temperature [K]
For each SolarBoat Design
SolarBoat alternatives
SolarBoat Design – A: Simulating driving
SolarBoat Design – B
Swap in an alternative design
Cumulative distribution of model output
1
0
Win probability
Consolidate Monte Carlo trials
Cost [¥]
Check over or under budget limit [¥ 150,000]
Chart line style: – Under budget ... Over budget
For each Monte Carlo trial
Speed [ms^{-1}]
Compute time to complete race
Time to complete race [hours]
Compute race win probability
Win probability
Race length 20 km
Win probability value function:
Probability of winning
Time to complete race [hours]

Fig. 58.1 Process overview

Modelica, which for this research is known as a "Level 0 Environment Interface" (red dashed box in Fig. 58.1), within which a common base class for all SolarBoat designs is created known as "Level 1 SolarBoat Interface." It is as if a socket has been created into which alternative SolarBoat designs can be placed (orange box in Fig. 58.1).

Then, we model the SolarBoat itself (i.e., the system of interest we are trying to design) from the common base class, hence utilizing a Modelica language feature that is well suited in comparing various different designs, as it is possible to enforce a particular interface and have variables common to all alternatives. This approach simplifies the creation and comparison of different designs that can have significantly different architectures. This is illustrated in Fig. 58.1 by the swapping in of various different SolarBoat alternatives.

We incorporate uncertainty by modeling external factors as probability density functions, which for SolarBoat are water current, solar irradiance, and ambient temperature. Using Monte Carlo simulation, we then compute the expected value of the system by consolidating the results of multiple trials into a cumulative probability distribution.

58.4 Application

58.4.1 Problem Description

The SolarBoat race occurs every year at the end of August on Lake Biwa in Japan, where different student teams race solar powered autonomous boats they have designed, built and tested over the previous semester. The competition involves 2 days of racing with the results of both days combined. On both days, the boats are challenged to travel 20 km on a predefined course with three waypoints (see as Fig. 58.2 left side). The race organizers set design constraints on the power train in the form of maximum solar panel size of 2m^2 and a maximum of 20 Wh of lead-based batteries (control systems can have additional batteries of any type).

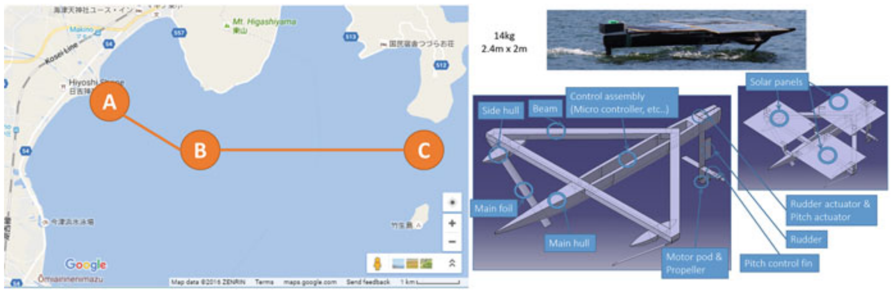


Fig. 58.2 *Left*: Map of the race route (A - > B - > C - > B - > A) (Modified from Google Maps [18]) *Right*: An example SolarBoat design (a hydrofoil from 2014) from the study by Sutherland et al. [19]

In addition, the boats are required to carry a payload from the race organizers. An example boat design is shown in the right side of Fig. 58.2.

To complete the project the University of Tokyo team typically has a budget of 300,000 Japanese Yen (¥), the final design's component bill typically incorporates no more than half of the budget (¥150,000), but students can make use of components from previous years for free. Given labor is free (students do the work), the sum of the component costs is an adequate model of the cost of the boat.

For this study, the boat is assumed to travel in a straight line for 20 km removing the need to model turning behavior. Further, the boat steady-state speed is assumed to be reached at a time equal to 100 seconds in the simulation and then assumed to remain constant (thus acceleration is not assessed). This velocity is used to calculate total time to complete the race, which is mapped to probability of winning the race by way of the value function described in Sect. 58.4.2. The environmental inputs (water current, solar irradiance and ambient temperature) that affect the boat are described in Sect. 58.4.3.

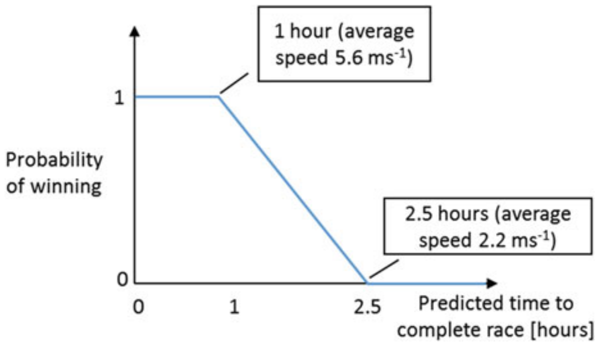
58.4.2 Value Function

The value function used in this work is shown in Fig. 58.3. It is believed that finishing the race in 1 h or less guarantees winning the race, while finishing in 2.5 h or more would guarantee losing. Winning is defined as finishing in first position in the race. Losing is defined as not winning. The bases for these two extremes are from limited data from previous races (which unfortunately does not specify the weather conditions experienced during the race), specifically:

- In 2015, Race Tokyo finished 2nd with 2 h 6 min.
- In 2010, Race Tokyo won the race and set the all-time race record with 1 h 40 min.

The probability of winning for finishing times between 1 and 2.5 h is believed to behave linearly. We have not elicited this function from existing data at this point, but we consider that it does not negatively affect the objectives of this paper.

Fig. 58.3 Win probability value function



Furthermore, other effects such as weather or sea conditions are considered to be compounded in the given value function. Future work is planned to show how the different conditions can be compounded to a single value function through Bayesian probabilities.

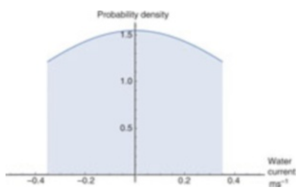
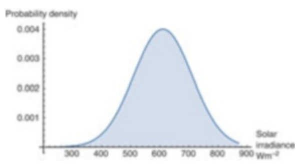
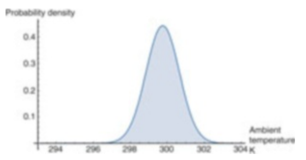
It should be noted that the actual competition consists of two races held on 2 days with the results combined. This has been ignored in this paper for simplicity, since it does not affect negatively the purpose of the paper.

However, unconstrained optimization of the designs with regard to probability of winning is insufficient as the project has a spending limit. Therefore, any designs that cost more than the ¥150,000 cost limit are considered infeasible and marked as such.

58.4.3 External Environment

Previous experiences in the SolarBoat race competition inform that water current, solar irradiance, and ambient temperature constituting the external environment drivers for boat performance in terms of finishing time. They have been modeled as probability density functions, in line with findings in literature. Their probabilistic models are provided in Table 58.1. While there are some dependencies between the

Table 58.1 Uncertainty models used

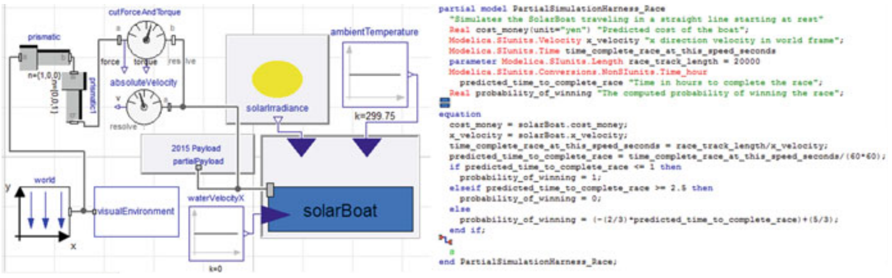
Environmental input	Probability distribution used		Description
Water current [ms^{-1}]		Truncated normal distribution: SD: 0.5Cut off: $-0.35\ ms^{-1}$Mean: $0\ ms^{-1}$Cut off: $0.35\ ms^{-1}$	Water currents maximum and minimum taken from the work of Endoh [20]. We assume due to long running processes even large lake currents are experienced
Solar irradiance [Wm^{-2}]		Truncated normal distribution: SD: 100Cut off: $260\ Wm^{-2}$Mean: $610\ Wm^{-2}$Cut off: $870\ Wm^{-2}$	Past work on the project had found the maximum and minimum solar irradiance Extreme values are not expected frequently
Ambient temperature [K]		Truncated normal distribution: SD: 0.9Cut off: 294.25 KMean: 299.75 KCut off: 306.35 K	Dry bulb temperature data from NOAA [21]

three elements, independence has been assumed for simplicity purposes in this paper. It should further be noted that wind and air resistance is ignored. This is justified as the density of water is 1000 times more than air and all SolarBoat designs are assumed to have a low-exposed area to the airflow (i.e., no sails or tall cargo); thus, hydrodynamic forces dominate aerodynamic forces. While these assumptions do reduce the accuracy of the models, they do not negatively affect the objectives of this paper.

58.4.4 System Model

To translate the problem description and assumptions into a computational model, initially a partial Modelica model was created with an interface defined for the SolarBoats to be placed and assessed. Left side of Figure 58.4 depicts this with a diagram view and right side shows the truncated Modelica code for the same model. The diagram view shows the interface for the SolarBoat clearly, while the truncated code shows that the assessment of the probability of winning is computed from the predicted time to complete the race, with the boats velocity and cost being extracted from the boat model.

Left side of Figure 58.5 shows the Modelica diagram view of the SolarBoat interface, showing that the engineer has much freedom to implement the design as they think best, but then consistently assess it with the model shown in Fig. 58.4. Figure 58.5 right side is a representation of the interface at Level 0, populated with different System of Interest designs (SolarBoats with different architectures, hence the different subsystems and subsystem components) and with SolarBoat designs being assigned Hierarchy Level 1 and Subsystems Hierarchy Level 2 and Subsystem-Components Hierarchy Level 3. As such, it is possible to assess all the alternative designs by the same method.



```
partial model PartialSimulationHarness_Race
  "Simulates the SolarBoat traveling in a straight line starting at rest"
  Real cost_money(unit="yen") "Predicted cost of the boat";
  Modelica.SIunits.Velocity x_velocity "x direction velocity in world frame";
  Modelica.SIunits.Time time_complete_race_at_this_speed_seconds
  parameter Modelica.SIunits.Length race_track_length = 20000
  Modelica.SIunits.Conversions.NonSIunits.Time_hour
    predicted_time_to_complete_race "Time in hours to complete the race";
  Real probability_of_winning "The computed probability of winning the race";

equation
  cost_money = solarBoat.cost_money;
  x_velocity = solarBoat.x_velocity;
  time_complete_race_at_this_speed_seconds = race_track_length/x_velocity;
  predicted_time_to_complete_race = time_complete_race_at_this_speed_seconds/(60*60);
  if predicted_time_to_complete_race <= 1 then
    probability_of_winning = 1;
  elseif predicted_time_to_complete_race >= 2.5 then
    probability_of_winning = 0;
  else
    probability_of_winning = (-(2/3)*predicted_time_to_complete_race)+(5/3);
  end if;

end PartialSimulationHarness_Race;
```

Fig. 58.4 Modelica model partial simulation harness race. Known as a “Level 0 Environment Interface”. *Left*: diagram view. *Right*: Text view (code is truncated for ease of reading)

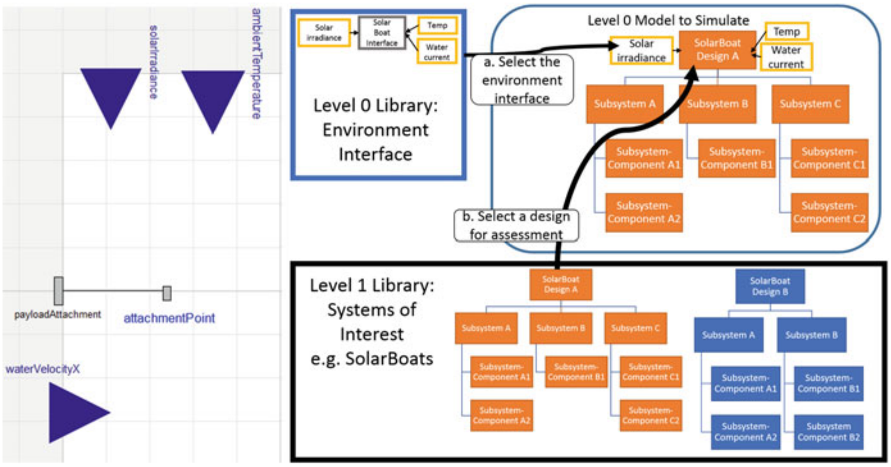


Fig. 58.5 *Left*: Modelica model of the "*Level 1* SolarBoat Interface". *Right*: Representation of combining SolarBoat designs (*Level 1*) with the Environmental Interface

58.4.5 Design Alternatives

Seven design alternatives are considered in this application (ref. Table 58.2). All the alternatives have the same subsystems of "Electrical to Thrust," "Buoyancy Generation," "Solar to Electrical," and "Overhead components." Thus, they all share the same architecture at Hierarchy Level 2. These subsystems are then decomposed into subsystem-components and specific designs are created by assigning specific subsystem-component implementations to the architecture. The resulting approach is similar to a morphological box technique, but with the synthesis of the functionality required of the subsystems being identified by the utilization of previous research by the authors which utilized OPM (Object Process Methodology) [4, 5]. The designs were selected to show a range of performances, not an optimally designed boat. Description of each component is provided in Table 58.3. In addition, all designs contain the same overhead components (control and structure). These are not displayed in Table 58.2 because of page limitations.

58.4.6 Simulation Conditions

100 Monte Carlo trials were run for each design, selecting environmental variables from probability distributions presented in Table 58.1. Simulation was conducted in Dassault Systèmes Dymola.

Table 58.2 Subsystem-components used in the designs. Descriptions of the components are provided in Table 58.3 Note that all designs contain both overhead components

			Subsystem to subsystem-component decomposition:				
			Electrical to thrust			Buoyancy generation	Solar to electrical
Design:	Cost (¥)	Mass (kg)	Motor	Gearbox	Propeller (mm)	Displacement hull	Solar panels
SB1	39,400	9.18	L3040A-480G	13:1	200	Single hull	Ft-136SE
SB2	39,400	9.18	L3040A-480G	13:1	220	Single hull	Ft-136SE
SB3	20,000	10.9	S13560_260R	3:1	220	Single hull	Ft-136SE
SB4	20,000	10.7	S13560_260R	None	160	Single hull	Ft-136SE
SB5	451,500	13.76	S13560_260R	3:1	220	Dual hull	SP50f
SB6	20,000	10.9	S13560_260R	3:1	160	Single hull	Ft-136SE
SB7	20,000	10.7	S13560_260R	None	220	Single hull	Ft-136SE

Table 58.3 SolarBoat designs used in this example

Component type	Name	Cost (¥)	Mass (kg)	Further details
Motor	L3040A-480G	2100	0.19	Low mass low torque (Kv = 480 rpm/volt)
	S13560_260R	0 (retail 202,500)	1.75	Retained from previous years. (Kv = 69 rpm/volt)
Gearbox	13:1	17,300	0.03	Compact planetary gearbox
	3:1	0	0.2	Made by students from existing parts
	None	0	0	For architectures with no gearbox
Propeller	220, 200 or 160 mm	0	0.02	Made by students. 220, 200 or 160 mm diameter, 2 blades
Hull	Single hull	20,000	0.69	Made by students but requires much material. 2.3 m × 0.17 m × 0.19 m
	Dual hull	40,000	1.39	Made by students but requires much material. 2.3 m × 0.17 m × 0.19 m
Solar panels	FT-136SE	0 (Retail 450,000)	3.24	Retained from previous years. Six panel array,13.5% efficient
	SP50f	411,500	5.4	Six panel array, 20.5% efficient
Overhead	Control	0	2.7	Control system. All boats assumed to have this. From previous years
	Structure	0	2.3	Miscellaneous extra mass. All boats assumed to have this

58.4.7 Results

The results of the assessment of the various designs are presented in Fig. 58.6. The left side plot shows the cumulative predicted time to complete the race. The right side plot shows the cumulative probability of winning. Designs found to be over the budget limit are marked by the fine dashed line (i.e., SB5 is too expensive). Because of the uncertainty associated with these results, SB3 was simulated for 10, 100, and 500 trials to test the effects of number of Monte Carlo trials. Figure 58.6 shows that when 10 trials are used, SB3's performance is significantly different than with 100 and 500 trials. The similarity of 100 and 500 trials indicates that 100 is sufficient. However, this experiment shows SB2 and SB3 are producing very similar performance and the difference in the simulation is likely uncertainty. As such, for designs with value predicted to be somewhat similar the engineer should exercise caution and investigate further.

Of the designs, SB1 strongly dominates the alternatives. To investigate SB1 further in comparison to SB2 and SB3, all three were simulated under mean environmental conditions (results in Table 58.4). It reveals that SB1 is the fastest despite not having the highest steady-state thrust. Thrust generated is a complex interaction between the spin speed of the power train and the boat velocity. Reviewing the thrust profiles (Fig. 58.7 left) shows SB1 has a wider thrust band enabling the boat to accelerate for a longer period of time. Reviewing the motor

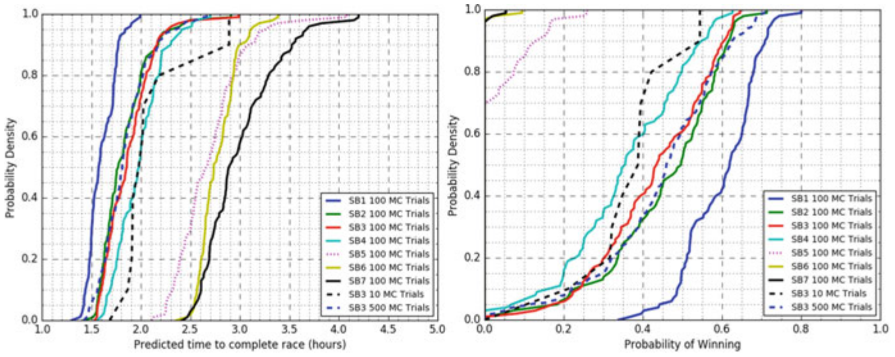


Fig. 58.6 Cumulative probability density functions for predicted time to complete race (*left side*) and probability of winning the race (*right side*) for multiple SolarBoat designs (Table 58.3). Note that the fine *dash line* indicates the design is beyond the budget that is SB5

Table 58.4 SolarBoat designs used in this example

	Results at 100 s				
Design	Velocity (ms^{-1})	Drag (N)	Thrust (N)	Motor nominal speed (rad s^{-1})	Motor speed (rad s^{-1})
SB1	3.5	16.9	16.9	949	1030
SB2	3.1	14.4	14.4	949	918
SB3	3.1	17.0	17.0	272	217

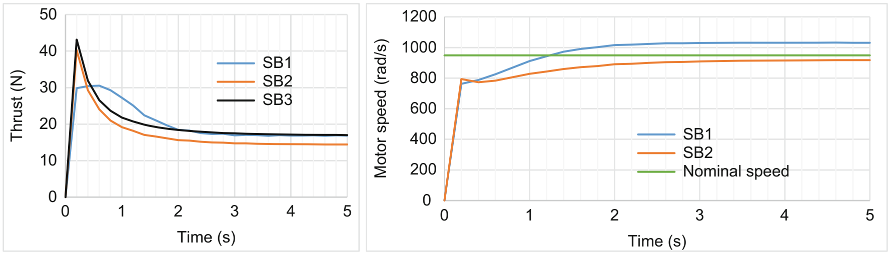


Fig. 58.7 *Left*: Thrust profiles for three solar boat designs. *Right*: Motor spin speed comparison

spin speeds (Fig. 58.7 right) shows SB1 is the only one to reach the nominal speed. Given SB1 scores so well, the importance of good power train matching is clear. In 2015, a design using SB2's drive train had been selected based on static thrust testing, and the research presented here indicates the smaller propeller of SB1 would be higher performance if the boat were moving.

58.5 Discussion

58.5.1 Benefits

The proposed approach makes it possible to logically compare and subsequently select alternative designs that are subject to uncertain environments by providing an assessment of the expected value created by each alternative and a measure of uncertainty associated with the said value measure. For the case study of the SolarBoat, this measure was the probability to win the race, for which cumulative probability distributions were created for each alternative design (Fig. 58.6 right side). This operationalization of VDD enables the project team to explicitly target the metric of success and model it explicitly (i.e., the value function displayed in Fig. 58.3 for the running case study), enabling the logical ranking of the performance of various designs. This contrasts with the authors' past work where either a multiobjective value function was used [4, 6] or the time to complete the race was predicted [5], which then created optimization problems that do not focus what the team is ultimately trying to achieve (i.e., win the race). By adopting VDD for this paper, the team can focus clearly on this value; thus, the expected benefits of VDD have been demonstrated.

Given the uncertainty present in the environment that the design operates in, it is critical to capture this to the best of our knowledge and use it to make informed decisions. The use of a Monte Carlo method and comparison of cumulative probability distributions provided a method to achieve this. Previous work by the authors simulated for a range of weather conditions but did not take into regard the likelihood for each of those weather conditions. This paper integrates the various weather conditions based on their individual likelihoods and thus provides an overall likelihood of a particular design to win the race.

Further, similar to the authors' previous studies, the use of the hierarchical object-oriented features of Modelica enabled the rapid comparison of alternative designs such that there can be encouragement to review significantly different designs (which can all utilize the same interface) and so consider designs which might otherwise not be considered. All of which can now be assessed against the primary value of winning the race.

58.5.2 Limitations

The models presented of the environment, race, and boats are likely not sufficient to provide the information needed to make all the decisions regarding what system architecture to build and race. Therefore, their fidelity should be improved. Further, the results output (Fig. 58.6 right side) lacks any assessment of the uncertainty that these results are subject to, making it potentially difficult to select between designs.

Finally, the synthesis process for the creation of design alternatives is only described briefly and needs to be expanded, particularly with regard to the flow down of the value function such that subsystems and subsystem components can be designed contributing to the overall project value.

58.6 Conclusions and Future Work

Given VDD is seen as a promising alternative to the inadequacies of the current design approaches, such as requirement-based design and Cost As Independent Variable (CAIV), the aim of this paper was to show how Modelica can be used to operationalize VDD and demonstrate its use on a novel case study. We applied the approach to the selection of a conceptual design for a student autonomous solar powered boat race, which given its novel nature, required a nonmonetary value function (probability to win the race) to be used. Modelica's object-oriented features were shown to enable the assessment of all the designs consistently and Monte Carlo simulation allowed the introduction of uncertainty.

While the approach was shown to provide much benefit for implementing VDD, there are limitations and obstacles identified that should be addressed in future work. We identified primarily four key topic areas of interest: the creation of higher fidelity models, uncertainty assessment of the results, the design synthesis approach, and value model decomposition. Each of these are addressed below.

First, the limitations of the models used in the research have been described in detail previously. Specifically, to address the environmental model, the independence of environmental variables should be removed and additional environmental variables introduced (e.g., waves and gusts of wind). This would then lead to more accurate modeling of the race of which a more accurate value model could be created making use of more data points of past race performances and incorporate the weather conditions to represent the beliefs the team has that the boat will win for a range of environmental conditions. As for the boat designs, currently the boat is not simulating turning functionality and thus not simulating the full journey between the waypoints. In addition, reliability performance is critical for project success and thus should be included in the model with breakdowns increasing the time to complete the race.

Second, an appreciation of the uncertainty in the results would be beneficial for decision-making (e.g., add *thickness* to the results plots in Fig. 58.6).

Third, this paper uses System of Interest models (i.e., SolarBoat designs) that were created in prior research; there is no attempt to provide an explicit methodology for the synthesis process (of function, system architecture, or parameters). Past work by Sutherland et al. [4] has attempted research in this area by means of the OPM language. However, further work is required as this offers an avenue to which this work can be generalized to other domains.

This leads to the forth topic for further work; value model decomposition, as the synthesis of designs will require such decomposition to enable the synthesis of a set of subsystems and subsystem-components which themselves address their own value functions derived from the whole system value function.

References

1. Blanchard BS (2008) System engineering management, 4th edn. Wiley, Hoboken
2. Collopy PD, Hollingsworth PM (2011) Value-Driven Design. J Aircr 48:749–759. doi:10.2514/1.C000311
3. Lee BD, Binder WR, Christiaan JJP (2014) A systematic method for specifying effective value models. Procedia Computer Science 28. 2014 conference on systems engineering research, pp 228–236. doi:10.1016/j.procs.2014.03.029
4. Sutherland J, Oizumi K, Aoyama K, Eguchi T, Naoki T (2016) System-level design tools utilizing OPM and modelica. In: ASME 2016 international design engineering technical conferences and computers and information in engineering conference IDETC2016. Charlotte
5. Sutherland J, Oizumi K, Aoyama K (2016) Design exploration of alternative configurations using 1D–CAE: a value-driven design approach utilizing OPM and modelica 「1D–CAEによる製品構成の探索」. In 26th Japanese Society of Mechanical Engineers (JSME) systems and design conference「日本機械学会第26回設本工学・システム部門講演会」. Keio University, Japan
6. Sutherland J, Oizumi K, Aoyama K, Takahashi N, Eguchi T (2016) System-level design trade studies by Multi Objective Decision Analysis (MODA) utilizing Modelica. In: The first Japanese modelica conference, May 23–24, Tokyo, Japan, pp 61–69. Tokyo: Linköping University Electronic Press
7. Hazelrigg GA (1998) A framework for decision-based engineering design. J Mech Des 120:653–658. doi:10.1115/1.2829328
8. Von Neumann J, Oskar M (1953) Theory of games and economic behavior, 3d edn. Princeton University Press, Princeton
9. Collopy PD (2012) A research agenda for the coming renaissance in systems engineering. In: 50th AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition, pp 2012–799
10. SEBoK (2015) Guide to the Systems Engineering Body of Knowledge (SEBoK)
11. Fritzson P (2010) Principles of object-oriented modeling and simulation with Modelica 2.1. John Wiley & Sons, Hoboken
12. Tiller M (2001) Introduction to physical modeling with modelica. Springer Science & Business Media, Boston
13. Tummescheit H (2002) Design and implementation of object-oriented model libraries using Modelica. PhD Theses TFRT-1063
14. Otter M, Elmqvist H, Mattsson SE (2003) The new modelica multibody library. In: 3rd International modelica conference, pp 311–330
15. Paredis Chris, Bernard Y, Burkhart R, de Koning H, Friedenthal S, Fritzson P, Rouquette N, Schamai W (2010) An overview of the SysML-Modelica transformation specification presented at the INCOSE symposium 2010, Chicago
16. Starling AC, Shea K (2005) A parallel grammar for simulation-driven mechanical design synthesis. pp 427–436. doi: 10.1115/DETC2005-85414
17. Du W, Garcia HE, Binder WR, Christiaan JJP (2014) Value-driven design and sensitivity analysis of hybrid energy systems using surrogate modeling. In: Renewable energy research and application (ICRERA), 2014 international conference on, 395–400. IEEE
18. Google Maps, Zenrin. Lake Biwa, Shiga Prefecture. Google Maps

19. Sutherland, J, Kamiyama H, Aoyama K, Oizumi K (2015) Systems Engineering and the V-Model: Lessons from an Autonomous Solar Powered Hydrofoil. In: 12th International marine design conference (IMDC). Tokyo
20. Endoh S (1978) Diagnostic analysis of water circulations in Lake Biwa. Journal of the Oceanographical Society of Japan 34:250–260. doi:10.1007/BF02111170
21. National Oceanic and Atmospheric Administration (NOAA) 1990 Hikone Weather Monitoring Station Data

Chapter 59
A Game Theoretical Perspective on Incentivizing Collaboration in System Design

Sean D. Vermillion and Richard J. Malak

Abstract Proponents of value-driven design hypothesize that multidisciplinary design optimization architectures can serve as templates for coordinating domain experts in the system design process. However, such architectures can rely on domain model sharing among the domain experts. In this paper, we do not take for granted that domain experts would provide these models. Therefore, we use game theory to formulate a mathematical model of agents' decisions to collaborate or not and analyze how we can formulate a piece rate incentive to motivate agents to collaborate. A piece rate incentive structure provides a marginal increase in reward with a marginal increase in some performance figure of merit. We provide a lower bound condition for the marginal increase in reward term in the piece rate structure that theoretically motivates an agent to collaborate. However, this lower bound is only reasonable if the agent believes that unilaterally collaborating produces an increase in performance.

Keywords Collaboration • Incentive formulation • Game theory

Nomenclature

CSSO	Concurrent subspace optimization
JIMS	Joint Simulation System
MDO	Multidisciplinary design optimization
VDD	Value-driven design
f_0	Performance figure of merit
K_1	Piece rate incentive marginal reward parameter

S.D. Vermillion (✉)
Innovative Decisions, Inc., Vienna, VA, USA
e-mail: svermillion@innovativedecisions.com

R.J. Malak
Department of Mechanical Engineering, Texas A&M University, College Station, TX, USA
e-mail: rmalak@tamu.edu

A.M. Madni et al. (eds.), *Disciplinary Convergence in Systems Engineering Research*, https://doi.org/10.1007/978-3-319-62217-0_59

K_2	Piece rate incentive guaranteed reward
U_i	Agent i's utility function
θ_i	Cost, penalty, or disutility of model sharing

59.1 Introduction

59.1.1 Motivation

This paper is about building foundational knowledge on incentivizing system designers to collaborate with each other during the system design process. The knowledge required for the design of large, complex systems is often far larger than any one individual's maximum knowledge capacity. Therefore, many domain experts, whose collective knowledge spans the requisite knowledge for system design, are needed. However, the success of a system design intuitively hinges on the ability of these domain experts to collaborate, coordinate, and share information about their specific design problems and decisions with each other. Naturally, the following question arises: how should these domain experts collaborate and coordinate in order to yield valuable systems? Proponents of value-driven design (VDD), a design paradigm that stipulates the objective of the design process to yield valuable designs [1], hypothesize that multidisciplinary design optimization (MDO) architectures can serve as templates for coordinating the design activities of many domain experts [1–4]. This hypothesis seems well founded as the central purpose of MDO architecture formulations is to optimize a particular objective function when we have several sources of information, i.e., discipline analysis models. In a VDD context, our objective function yields some figure of merit for the system value, and coordinating domain experts according to some MDO architecture intuitively supports discovering design solutions that optimize system value.

Much of the MDO research is concerned with the mathematical and numerical properties of MDO architectures. Specifically, MDO research seeks to uncover a particular problem formulation's ability to optimize a system objective function as well as benchmarking the computational costs associated with solving an MDO problem with a particular MDO architecture. However, there is less consideration for how human decision-making impacts the success of an MDO architecture. This consideration is rather important when we are considering an MDO architecture as a template for coordinating domain experts as opposed to merely a computational structure for solving a design problem. As independent agents, domain experts in the system design problem are endowed with *agency*, meaning they have the ability to make their own decisions. Therefore, domain experts in the design process have the ability to decide whether or not they want to abide by a set design protocol. For example, the concurrent subspace optimization (CSSO) architecture relies on sharing information about each discipline's analysis model [5], but a domain expert may choose not to share his model information for one reason or another. The research presented in this paper is particularly focused on this scenario and uncovering promising incentive structures to motivate a domain expert to want to share his model information.

For this research, we are essentially not taking collaboration between domain experts in the form of model information sharing for granted. To motivate this position, consider the following cases. Austin-Breneman et al. studied student design teams working toward designing a satellite system with three subsystems, and each student in a team had at most only information relevant to a single subsystem [6]. They observed that, despite being given a tool to share model information in the form of gradients and having at least one member of the team familiar with MDO, the students largely focused on designing their individual subsystems without much communication of this model information to the other team members. In a different case, Barry and Koehler examine the failures of the Joint Simulation System (JSIMS), a war simulation engine that would be built by joining the war simulation models developed by each of the US service branches [7]. They note that the JSIMS program lacked incentives for the branches to collaborate, and as participants dropped out, the chance another would drop out of the program increased. Finally, Witus notes that vendors in a US Army advanced technology demonstration program had proprietary models they were reluctant to make available for integration [8].

From the example cases above, we see that (1) collaboration in the form of model sharing is not guaranteed, (2) an agent's decision to collaborate depends on whether other agents collaborate, and (3) collaboration in the form of model sharing can accrue some penalty or cost. In this paper, we use the mathematical power of game theory to model and analyze agents' decisions to collaborate in response to a performance incentive while considering the three assumptions listed above. A performance incentive is one in which the reward is based on the performance of a final product rather than the process used to create it [9]. Using the mathematics of game theory, we define the theoretical conditions where a performance incentive motivates a rational agent to collaborate with their model information in the system design process. The goal is to generate qualitative conclusions from the mathematical theory so we can if an MDO architecture that requires model sharing would be successful for system value maximization.

This paper is structured as follows. The remainder of this section motivates why we are interested in performance incentives for motivating collaboration as well as provides background on modeling collaboration in systems engineering. Next, the foundations of game theory are summarized to provide the basis for the modeling and analysis activities in this paper, which are presented in Sects. 59.3 and 59.4. The paper concludes with a discussion and summary of the contents herein.

59.1.2 Incentives Types

There are two general incentive types [9]: (1) incentives that give out a reward based on performance and (2) incentives that give out a reward based on behavior. To differentiate the two, consider an academic situation. Students receive points based on how well they answer exam and homework problems, i.e., how they perform on exams and homework. Students may also receive points by participating in in-class discussions. We would say the first case reflects a performance-based incentive since we are concerned with quality, e.g., quality of the exam answers. We would say the second case reflects a behavior-based incentive since we are concerned with behavior, e.g., students engaging in in-class discussion. UPS uses both behavioral and performance incentives for its drivers by not only monitoring and rewarding the volume and expediency of package delivery (performance) but also monitoring and rewarding driving habits (behavior) [10].

An important question for incentive formulation in the context of collaborative model sharing is *why not penalize an agent if they do not model share?* Essentially, if an agent does not model share, they accrue a penalty, or if penalties are not feasible, an agent receives a bonus if they model share. This type of incentive constitutes a behavioral incentive structure since we are directly trying to control behavior, i.e., model sharing behavior, with this type of structure [9]. The problem with a behavioral incentive structure is that it requires some kind of verifiability. This is to say that we have to verify that an agent indeed model shares, which might seem trivial by itself, but we would also care that the agent is sharing a useful model, which is more difficult to do if we are not experts in the model's particular domain. With a performance-based incentive structure, an agent would theoretically supply the information he thinks would yield the best performance, thus yielding greater incentive payout. Therefore, we consider a performance-based incentive for the analysis in this paper. The particular performance incentive formulation under consideration in this paper is given in Sect. 59.3.1.

59.1.3 Modeling Collaboration

In this paper, we use game theory to generate a first-order model of collaborative model sharing that incorporates the following concepts: (1) collaboration in the form of model sharing is not guaranteed, (2) an agent's decision to collaborate depends on whether other agents collaborate, and (3) collaboration in the form of model sharing can accrue some penalty or cost. However, prior research has used mathematical models to investigate collaboration in other contexts. Takai uses game theory to model and investigate how engineers working on a product platform should allocate their time between working on the platform itself and their individual platform modules [11]. Lewis and Mistree formulate independent models for the case where domain engineers share their domain information and the case where domain engineers do not share their domain model information but rather share their design variable best responses to the decisions of the other domain experts [12, 13]. In this paper, we differentiate ourselves from Lewis' and Mistree's work in that we are endowing our agents the ability to choose if they want to share their domain model information or not.

Arsenyan et al. use game theory to model and analyze a case where two firms collaborate for a common product [14]. In this model, the firms share the revenue generated by the product and choose how much information and trust to give to the other firm. Arsenyan et al. use this model to derive the optimal amount of information sharing between firms. Our aim differs from that of Arsenyan et al. in that we are concerned with formulating an incentive that induces collaborative information sharing between agents. Therefore, we are essentially designing the game to produce a specific game solution, i.e., Nash equilibrium as discussed below, as opposed to deriving the solution to a given game.

59.2 Game Theory Foundations

Game theory is an extension of decision theory that studies conflicts between rational decision-makers [15]. The strategic interaction between decision-makers is termed a game and is the base unit of interest in game theory. The simplest game structure is the *normal form* game. In a game in normal form, we have a set of players, $\mathbf{I} = \{1, \ldots, i, \ldots, N\}$, and an N-tuple of strategy sets, $\mathbf{S} = \{\mathbf{S}_1, \ldots \mathbf{S}_i, \ldots, \mathbf{S}_N\}$, where $\mathbf{S}_i$ is the set of strategies available to player i. Additionally, $\mathbf{U} = \{U_1, \ldots, U_i, \ldots, U_N\}$ is an N-tuple of utility functions, one for each player in the game, such that $U_i : \prod_{j \in N} \mathbf{S} \rightarrow \mathbb{R}$. To exemplify a game in normal form, consider the prisoner's dilemma game in Fig. 59.1. Here, each player, or prisoner, has two strategies: (1) cooperate with the other prisoner or (2) defect and betray the other prisoner. Additionally, each prisoner has a certain utility that is dependent on the strategies of each prisoner. Prisoner 1's utility is given in the lower left corner of each game cell, and Prisoner 2's utility is given in the upper right corner of each game cell.

		Prisoner 2	
		Cooperate	Defect
Prisoner 1	Cooperate	−1 −1	0 −3
	Defect	−3 0	−2 −2

Fig. 59.1 The prisoner's dilemma game in normal form

The basic solution concept to a game between rational decision-makers is a Nash equilibrium, wherein no player can increase their utility by only changing their own strategy. Let $s_i \in \mathbf{S}_i$ denote a particular strategy in player i's strategy set; A strategy profile $\mathbf{s}^* = \{s_1^*, \ldots, s_i^*, \ldots, s_N^*\}$ is a Nash equilibrium if the following holds:

$$U_i(s_i^*, \boldsymbol{s}_{-i}^*) \geq U_i(s_i, \boldsymbol{s}_{-i}^*) \forall s_i \in \boldsymbol{S}_i \;\; \forall i \in \mathbf{I} \tag{59.1}$$

where $\boldsymbol{s}_{-i}^* = \boldsymbol{s}^* \setminus \{s_i^*\}$. In the prisoner's dilemma game, both prisoners choosing to defect is the Nash equilibrium since defecting dominates cooperating, i.e., each prisoner will always receive a higher utility from defecting despite what the other prisoner chooses. A Nash equilibrium is guaranteed to exist [16], and we will use the Nash equilibrium solution concept when we evaluate our performance incentive formulation in the following sections.

59.3 Model Formulation

59.3.1 Incentive Formulation

Here, we present our incentive structure under consideration, but let us first motivate their formulation with the concurrent subspace optimization (CSSO) MDO architecture used in a VDD context. A high-level, generic formulation of a CSSO discipline optimization problem is the following [5]:

$$\underset{x}{\text{maximize}} \;\; f_0(x, y_i(x, \tilde{y}_{-i}), \tilde{y}_{-i}) \tag{59.2}$$

where f_0 is the system-level objective function, x represents design variables, y_i represents the discipline i analysis model, and $\tilde{y}_{-i}$ represents some analysis model, e.g., surrogate model, from other disciplines. The incentive structure under consideration in this paper is rather intuitive. While referencing Eq. 59.2, we might offer a marginal increase in reward with a marginal increase in system value, f_0, such that the incentive structure takes the following form:

$$v_i = K_1 \times f_0 + K_2 \tag{59.3}$$

where $K_1 > 0$ is the marginal increase in reward payout and K_2 is some fixed, guaranteed payout. This incentive structure reflects the *piece rate* incentive structure formulation [17]. Maximizing f_0 in Eq. 59.3 maximizes the incentive payout since v_i^{PR} is simply a positive affine transformation of f_0. Assuming that sharing model information, i.e., sharing $\tilde{y}$, supports a greater system objective value, the incentive formulation above provides an incentive for an agent to share his model information. However, agents may not only care about maximizing their reward payout, but they may also care about minimizing any costs they might accrue while maximizing their reward payout. In the next section, these considerations are taken into account while formulating a model for collaboration decisions.

59.3.2 Collaboration Game Formulation

While agents seemingly have an incentive to collaborate to find an optimal system design – therefore maximizing their incentive reward – we now model the case where collaboration is costly. First, we model the collaboration model at a high level as a normal form game with N agents indexed in $\mathbf{I} = \{1, 2, \ldots, N\}$. Each agent has the option to collaborate with their model information, $s_i = 1$, or not, $s_i = 0$, such that agent i's strategy space is $\mathbf{S}_i = \{1, 0\}$, and $\mathbf{S} = \{\mathbf{S}_1, \mathbf{S}_2, \ldots, \mathbf{S}_N\}$ is an N-tuple of strategy sets. Finally, as with any decision model, agent i has a utility function U_i representing his preferences with $\mathbf{U} = \{U_1, U_2, \ldots, U_N\}$ being the set of utility functions in the game. Therefore, the collaboration game has the structure $\Gamma = \langle \mathbf{I}, \mathbf{S}, \mathbf{U} \rangle$. A graphical example of this game is shown in Fig. 59.2. It is important to note that not sharing model information, $s_i = 0$, does not necessarily mean that agent i does not participate in the design process. However, this can represent the case where agent i merely shares static information about his subsystem, e.g., sharing a static value for mass as opposed to some model of how another agent can influence the mass of agent i's subsystem; see Fig. 59.3.

Agent i's decision model related to collaboration is the following:

$$\underset{s_i \in \boldsymbol{S}_i}{\text{maximize}} \quad E[U_i(v_i(s_i, s_{-i}) - c(s_i))] \tag{59.4}$$

where $v_i(\cdot)$ is the incentive structure given to the agent and $c(\cdot)$ characterizes the penalties or costs accrued by the agent due to his collaboration decision. Intuitively, the incentive payout given to agent i depends not only on whether he collaborates with his model but also if the other agents collaborate with their models. However, the collaboration cost term only depends on agent i's decision. The cost model is as follows:

$$c(s_i) = \begin{cases} \theta_i & \text{\textit{if} } s_i = 1 \\ 0 & \text{\textit{if} } s_i = 0 \end{cases} \tag{59.5}$$

This model reflects that collaborating comes with a penalty or cost $\theta_i > 0$ and assumes that collaborating is much costlier than the alternative.

		Agent 2	
		$s_2 = 1$	$s_2 = 0$
Agent 1	$s_1 = 1$	$U_2(1,1)$ $U_1(1,1)$	$U_2(1,0)$ $U_1(1,0)$
	$s_2 = 0$	$U_2(0,1)$ $U_1(0,1)$	$U_2(0,0)$ $U_1(0,0)$

Fig. 59.2 The collaboration game between two agents in normal form

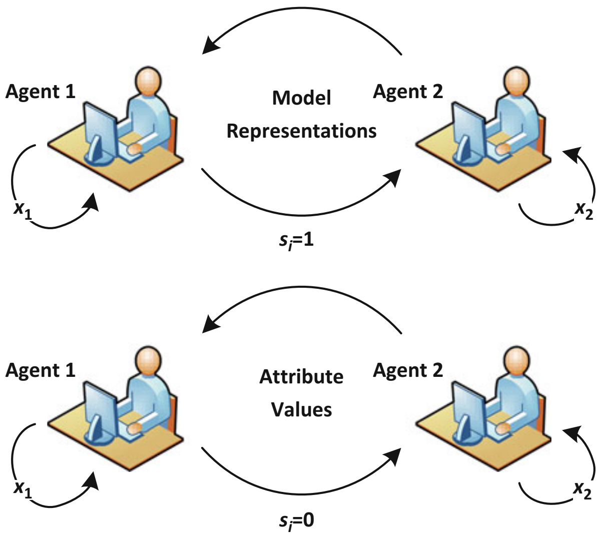


Fig. 59.3 The interpretations of an agent's strategy space

Since we are only concerned with a specific formulation of v_i in this paper, it is helpful to reformulate the decision model above into terms of Eq. 59.3. With the piece rate structure, agent i's expected utility is the following:

$$E[U_i(v_i(s_i, s_{-i}) - c(s_i))] = U_i(K_1 \times f_0(s_i, s_{-i}) + K_2 - c(s_i)) \tag{59.6}$$

The expectation operator is removed here since *ex post* reward is certain with the piece rate. Note the abuse of notation where $f_0(s_i, s_{-i})$ simply represents the fact that the *ex post* system objective value depends on with the agents share model information. The utility formulation in Eq. 59.6 can then be used to populate the utility terms in the collaboration game in Fig. 59.2.

59.4 Model Analysis

In this section, we analyze the collaboration game model to determine when the piece rate incentive structure in Eq. 59.3 induces an agent i to choose to collaborate with the model information. This is to say, we wish to find a way to set the incentive parameters K_1 and K_2 to ensure that agent i chooses $s_i = 1$. For simplicity, let us consider the perspective of Agent 1 in Fig. 59.2 without loss of generality. For Agent 1 to choose $s_1 = 1$, the following conditions must be true:

$$\begin{aligned} U_1(K_1 \times f_0(1,1) + K_2 - \theta_1) > U_1(K_1 \times f_0(0,1) + K_2) \\ U_1(K_1 \times f_0(1,0) + K_2 - \theta_1) > U_1(K_1 \times f_0(0,0) + K_2) \end{aligned} \tag{59.7}$$

The first condition ensures that Agent 1 will choose $s_1 = 1$ if Agent 2 chooses $s_2 = 1$, and the second condition ensures that Agent 1 will choose $s_1 = 1$ if Agent 2 chooses $s_2 = 0$. A reasonable assumption is that an agent's utility is monotonically increasing with its argument, i.e., utility increases with increasing "net reward." Therefore, the conditions above can be restated as the following:

$$\begin{aligned} K_1 \times f_0(1,1) + K_2 - \theta_1 > K_1 \times f_0(0,1) + K_2 \\ K_1 \times f_0(1,0) + K_2 - \theta_1 > K_1 \times f_0(0,0) + K_2 \end{aligned} \tag{59.8}$$

Through simple algebraic manipulation, these conditions become the following:

$$\begin{aligned} K_1 > \frac{\theta_1}{f_0(1,1) - f_0(0,1)} \\ K_1 > \frac{\theta_1}{f_0(1,0) - f_0(0,0)} \end{aligned} \tag{59.9}$$

This new formulation of the necessary conditions to ensure $s_1 = 1$ places lower bounds on K_1 parameter. However, we are only interested in the maximum lower bound since choosing a K_1 value that is greater than the maximum lower bound necessarily satisfies the other lower bound. Therefore, we are left with a necessary condition on choosing K_1 to ensure $s_1 = 1$ such that

$$K_1 > \max\left(\frac{\theta_1}{f_0(1,1) - f_0(0,1)}, \frac{\theta_1}{f_0(1,0) - f_0(0,0)}\right) \tag{59.10}$$

There is a similar condition for choosing K_1 in Agent 2's incentive structure.

The lower bound condition in Eq. 59.10 is dependent on an agent's cost, θ_i for sharing model information as well as the perceived increase in the system objective function value between sharing and not sharing model information, $f_0(s_i = 1, s_{-i}) - f_0(s_i = 0, s_i)$ for all possible cases of s_{-i}. Understandably, if there is no cost, i.e., $\theta_i = 0$, then the marginal reward just has to be positive such that $K_1 > 0$. However, when there is some cost, i.e., $\theta_i > 0$, the lower bound threshold for K_1 depends largely on the difference in perceived rewards due to sharing and not sharing, i.e., $f_0(s_i = 1, s_{-i}) - f_0(s_i = 0, s_i)$. If there is some s_{-i} where $f_0(s_i = 1, s_{-i}) - f_0(s_i = 0, s_i) = 0$, then the lower bound for K_1 is infinity meaning that there is no value for K_1 that guarantees that $s_i = 1$ is the dominant choice for agent i. Therefore, there is a chance that agent i would not share his model information in this case. This case might be a real concern. Consider again the JSIMS case where the success of the program seemingly hinged on *all* of the US military branches contributing their wartime simulation engines [7]. Here, there may be no perceived benefit in f_0 if agent i shares his model information when another agent does not share such that $f_0(1,0) - f_0(0,0) = 0$.

59.5 Discussion

In the previous section, we established a theoretical condition for setting a marginal reward, K_1, in a piece rate incentive structure such that an agent should normatively choose to collaborate with his domain knowledge, i.e., model, information despite what other agents do. From this theoretical condition, we can draw out some qualitative observations. Firstly, the guaranteed incentive payout K_2 does not influence whether an agent chooses to model share or not. Therefore, we cannot depend on tuning this parameter, e.g., offering more money in a contract that is not contingent on performance, to encourage model sharing.

The important parameter for promoting collaboration is the marginal reward parameter, K_1, and this parameter is linked to performance. However, there is only a real value of K_1 that guarantees collaboration (theoretically) if there is a perceived performance benefit from collaborating despite whether or not any other agent collaborates or model shares. If there is just one case where there is no perceived performance benefit, then there is no guarantee that the given agent will model share. Therefore, it may come to the contracting agency conveying to a given agent, domain expert, or vendor that unilateral model sharing inherently produces better performance, and thus higher incentive reward, than not doing such sharing. We show in the theoretical condition in Eq. 59.10 that if there is some performance benefit to unilateral collaborative model sharing such that $f_0(s_i = 1, s_{-i}) - f_0(s_i = 0, s_i) > 0$ for all s_{-i}, then there is some theoretical, real valued marginal reward, K_1, in a piece rate incentive structure that induces collaborative sharing.

Returning back to the VDD hypothesis that MDO architectures can serve as templates for coordinating domain experts in the system design process, we show that it is possible to induce collaboration with a performance incentive. Therefore, an MDO architecture that depends on higher order information sharing, like CSSO, might be successful as a template for system design. As stated above, however, this success depends on whether a given agent believes that unilateral collaboration will produce greater performance.

59.6 Summary

In this paper, we question whether we can take for granted that engineers would provide the information necessary to successfully use an MDO architecture as a template for conducting system design. We use the mathematical power of game theory to formulate a model of multi-agent collaboration and analyze how a piece rate incentive should be formulated to ensure the Nash equilibrium of the collaboration model lies at all agents choosing to collaborate. We discover a condition that theoretically guarantees that an agent will collaborate but also show when this condition can fail. The obvious limitation of this research is the validity of the theoretical model used to build insight into designing the piece rate incentive structure to encourage collaborative model sharing. Particularly, the model may be missing important terms and elements that would affect how we should formulate incentives. Future research in this context is specifically geared toward identifying these terms and elements, if they exist. Essentially, this future research seeks to validate the theoretical claims made in this paper.

Acknowledgments This material is based upon work supported by the National Science Foundation under Grant No. CMMI-1333454. Opinions expressed in this paper are of the authors and do not necessarily reflect the views of the National Science Foundation.

References

1. Collopy PD, Hollingsworth PM (2011) Value-driven design. J Aircr 48(3):749–759
2. Collopy PD, Bloebaum CL, Mesmer BL (2012) The distinct and interrelated roles of value-driven design, multidisciplinary design optimization, and decision analysis. In: 14th AIAA/ISSMO Multidisciplinary analysis and optimization conference, 2012
3. Bloebaum CL, Collopy PD, Hazelrigg GA (2012) NSF/NASA Workshop on the design of large-scale complex engineered systems – from research to product realization. In: 14th AIAA/ISSMO Multidisciplinary analysis and optimization conference, paper no. AIAA2012–5572, Indianapolis, 2012
4. Collopy P (2012) A research agenda for the coming renaissance in systems engineering. In: American Institute of Aeronautics and Astronautics Symposium, Reston, 2012
5. Martins JRRA, Lambe AB (2013) Multidisciplinary design optimization: a survey of architectures. AIAA J 51(9):2049–2075
6. Austin-Breneman J, Honda T, Yang MC (2012) A study of student design team behaviors in complex system design. J Mech Des 134:124504
7. Barry P, Koehler M (2013) Multiparty engineering is a contact sport. In: Systems conference (SysCon), 2013 I.E. International, IEEE, 2013
8. Witus G (2016) SERC TALKS: what lives at the intersection of MOSA (modular open system architecture) and set-based design? Available from http://www.sercuarc.org/events/serc-talks-what-lives-at-the-intersection-of-mosa-and-set-based-design-modular-open-system-architecture/. Accessed 19 Oct 2016
9. Eisenhardt KM (1989) Agency theory: an assessment and review. Acad Manag Rev 14 (1):57–74
10. Goldstein J (2016) The future of work looks like a UPS truck. Available from http://www.npr.org/sections/money/2014/05/02/308640135/episode-536-the-future-of-work-looks-like-a-ups-truck. Accessed 15 Jun 2016
11. Takai S (2010) A game-theoretic model of collaboration in engineering design. J Mech Des 132(5):051005–051005
12. Lewis K, Mistree F (1997) Modeling interactions in multidisciplinary design: a game theoretic approach. AIAA J 35:1387–1392
13. Lewis K, Mistree F (1998) Collaborative, sequential, and isolated decisions in design. J Mech Des 120:643–652
14. Arsenyan J, Büyüközkan G, Feyzioğlu O (2015) Modeling collaboration formation with a game theory approach. Expert Syst Appl 42(4):2073–2085
15. Barron EN (2013) Game theory: an introduction, vol 2, Wiley, Hoboken, New Jersey
16. Nash J (1951) Non-cooperative games. Ann Math 54(2):286–295
17. Bonner SE et al (2000) A review of the effects of financial incentives on performance in laboratory tasks: implications for management accounting. J Manag Account Res 12(1):19–64

Part VIII
Systems Engineering and Smart Manufacturing

Chapter 60
Toward a Diagnostic and Prognostic Method for Knowledge-Driven Decision-Making in Smart Manufacturing Technologies

Thomas Hedberg, Allison Barnard Feeney, and Jaime Camelio

Abstract Making high-quality manufacturing decisions in real-time is difficult. Smart manufacturing requires sufficient knowledge be available to the decision maker to ensure the manufacturing system runs efficiently and effectively. This paper will present background information for managing and controlling decision-making and technological innovation. We present a process definition for decision-making that implements closed-loop diagnostic and prognostic control. Lastly, we discuss our emerging concept relative to smart manufacturing.

Keywords Decision-making • Smart manufacturing • Technological innovation

60.1 Introduction

Smart manufacturing cannot be successful without proper management and technological innovation. The Oxford English Dictionary [1] defines technology as "the application of such knowledge for practical purposes." Innovation [2] is defined as "the alteration of what is established by the introduction of new elements or forms." And, management [3] is defined as "organization, supervision, or direction."

Using these definitions, we may define technological innovation as the process for creating a new application of practical knowledge. Thus, the management of technological innovation is the organization, supervision, or direction of the process for creating a new application of practical knowledge.

T. Hedberg (✉) • A. Barnard Feeney
Systems Integration Division, National Institute of Standards and Technology, Gaithersburg, MD 20899, USA
e-mail: tdh1@nist.gov; abf@nist.gov

J. Camelio
Grado Department of Industrial and Systems Engineering, Virginia Tech, Blacksburg, VA 24061, USA
e-mail: jcamelio@vt.edu

A.M. Madni et al. (eds.), *Disciplinary Convergence in Systems Engineering Research*, https://doi.org/10.1007/978-3-319-62217-0_60

While, several "smart" technologies have existed in manufacturing since the 1980s, the integration of those technologies along with the convergence of information technology (IT) and operational technology (OT) has kicked off a period of an increased rate of innovation in manufacturing. In general, Tidd and Bessant [4] presented "key lessons learned about managing innovation." Tidd and Bessant [4] recommended that organizations be visible in promoting innovation across the whole business, build a project-based organization with a good portfolio management structure, utilize a stage-gate system, and institutionalize the use of tools. We must remember innovation requires the creation of something new. Therefore, creativity, development processes, and change management must be accounted for in decision-making within the overall technological-innovation process.

Collaborative product development (CPD) [5], concurrent engineering [6], designed for manufacturing (DFM) [7], design for six sigma (DFSS) [8], and integrated product and process development (IPPD) [9] are popular business strategies for managing new-development activities. Decision-making is a common function in all of these strategies. Companies may combine these popular strategies with stage-gate processes to form their complete operating models. Further, industry desires to couple these methods with model-based systems engineering (MBSE), the "vee" diagram, and the larger-scoped model-based enterprise concept to enable effective decision-making during development and manufacturing processes [10].

However, organizations often apply these methods without ever re-asking if the development and manufacturing activities are still the right pursuits – that is, should the organization's overall goals change during and throughout the activities? This question and the desire to ensure the optimality, stability, effectiveness, and efficiency of technological-innovation process motivated this paper.

In this paper, we present our emerging and beginning work toward a diagnostic and prognostic method for knowledge-driven decision-making in smart manufacturing technologies. We will show that decision-making, technological innovation, and the management/control of both are not mutually exclusive. First, we provide background knowledge discussing decision-making, technological innovation, and the management of both, while also comparing various types of control theories (e.g., controls engineering, management control, and human factors). Then, we will present a process definition for decision-making and discuss the relationship between technological innovation, its management, and smart manufacturing. We will use this information to describe the beginnings of a concept for implementing closed-loop diagnostic control in technological-innovation and decision-making processes. Next, we will analyze and discuss our emerging concepts in relation to the "Digital Thread" in the manufacturing domain. Lastly, we will conclude with the utility of the concepts in supporting efficient and effective decision-making.

60.2 Background

While developing our concept for controlling the manufacturing decision-making process, we had to collect cross-discipline understanding of technological innovation because the various roles (e.g., marketing, engineering, management, and finance) might affect manufacturing decisions. We focused our research on three areas of understanding. The first focus area was in defining the technological-innovation process. The next focus area was in managing decisions for creativity, development projects, and changes in organizations. The final focus area was on control theories in the context of engineering, manufacturing, and management interactions.

60.2.1 Defining the Technological-Innovation Process

Knight [11] proposed technological innovation means an organization has adopted a new concept beyond the generation stage of the concept. Porter [12] suggested technological innovation is a "new way of doing things that is commercialized." Freeman and Soete [13] said, "an innovation in the economic sense is accomplished only with the first commercial transaction involving new product, process system, or device..." Tidd and Bessant [4] agreed innovation is the process of growing inventions into practical use. A diagram of the technological-innovation process based on Hollen [14] is shown in Fig. 60.1. The literature [14, 15], both recent and past, shows technological innovation as a three-step process of discovery, development, and deployment.

The first phase in the technological-innovation process is discovery. We may consider this phase synonymous with invention. New knowledge is created during the discovery phase. The output from the discovery phase is typically a conceptual design from a Research and Development (R&D) activity.

The second technological-innovation phase is development. This phase is a transition activity. In product development, the conceptual-design task is transitioning toward detailed-design activities. Management of technological innovation is important during the development phase because successful commercialization depends on the maturity level of the technology. The output of the development phase is a complete definition for the technology.

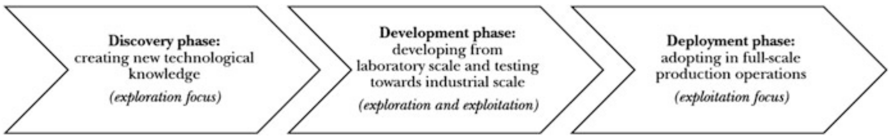

Discovery phase:
creating new technological knowledge
(exploration focus)
Development phase:
developing from laboratory scale and testing towards industrial scale
(exploration and exploitation)
Deployment phase:
adopting in full-scale production operations
(exploitation focus)

Fig. 60.1 Three-phase process definition for technological innovation (based on [14])

The third phase is deployment. This phase is where a process is being deployed to production operations, or products are available for delivery to the marketplace. Development is complete or near completion when the deployment phase begins. The output of the last phase is a new and complete technology.

Management must remain a critical focus during the deployment phase because many scholars consider the commercialization of technology the least managed activity in the technological-innovation process [16]. The methods used to commercialize and market technology significantly influence the success or failure of products [17]. Products with newly commercialized technology fail at a rate of 40–50% [16]. The demonstrated importance of management and decision-making is the motivation behind this paper.

60.2.2 Managing Decisions for Creativity, Development, and Change

"Creativity is the production of novel and useful ideas in any domain" [18]. Amabile [18] proposed a model of creativity that requires abilities in three major components, which are expertise, creative thinking, and intrinsic task motivation. The combined skills in each category enable creativity. The field of psychology teaches that anyone is capable of creativity, but the level of creativity is enhanced or limited by interactions with the social environment.

Lewin's Equation [19], $B = f(P, E)$, proposed behavior (B) is a function of interactions between people (P) and their environment (E). Following this idea, we argue innovation is a function of a person's creative ability and his/her interaction with the social environment. Further, Hoegl, and Parboteeah [20] suggested that the quality of team collaboration influences the utilization of the teams' technical skills and directs those skills toward the critical-performance dimensions.

Considering Hoegl and Parboteeah [20], we propose extending Lewin's Equation [19] to organizations by arguing that innovation is a function of the organization's overall creative ability and its social interactions within the environment. That is, $I = f(\sum P_i, E \in O)$, where I represents innovation, i represents individuals in the organization, and O represents the organization. Therefore, managing and encouraging creativity at the personal level should support a positive environment for innovation at the organizational level.

Amabile [18] argued that individuals with basic capacities can develop moderately creative solutions to some problems some of the time. However, challenging problems of high importance require subject matter experts with extensive knowledge in the field of work. A baseline level of expertise in the engineering domain is needed to ensure the ideas produced by the creative process are "novel and useful" [18].

Amabile's [18] and Hoegl's and Parboteeah's [20] conclusions support Cooper's [21] recommendations for including all critical roles in a product-development process from the start of the process. Cooper further suggests there are two ways to succeed in innovation – (1) doing projects right and (2) doing the right projects. Doing projects right requires a process to follow commonly accepted management guides. These guides should include using teams effectively, doing up-front research before starting development, analyzing the voice of the customer, and ensuring a stable product definition prior to deployment or launch. Doing the right projects requires the "right" expertise to know what the right portfolio of projects looks like. This relates to Amabile's [18] conclusion that a basic level of expertise is needed to determine if something is "novel and useful."

Cooper also developed a stage-gate process model that breaks the product-innovation process into five stages, each requiring the passage of a gate before proceeding to the next stage. The gates provide quality control to the process by incorporating go/no-go decisions at strategic points in the process. While Cooper's model provides a good foundation for managing product-development activities, it may fall susceptible to disruptive changes that could occur during the activities – specifically changes due to the technological-innovation process. This opens up Cooper's model to the risk of pursuing decisions that are no longer the right decisions.

Manufacturing organizations operate in an environment of constant change. Organizations must be prepared to manage the changes through effective decision-making. Managing changes effectively is an important part of ensuring sustainable success within an organization. Organizational strategies, structures, skills, and cultures must evolve over time to reflect changes in markets and technology [22].

Specifically related to technology, change happens in cycles [22]. These cycles are best explained with an illustration presented in Fig. 60.2. Technology cycles begin with high rates of innovation until a dominant technology emerges. As technology matures, the rates of innovation slow. As competition continues in the

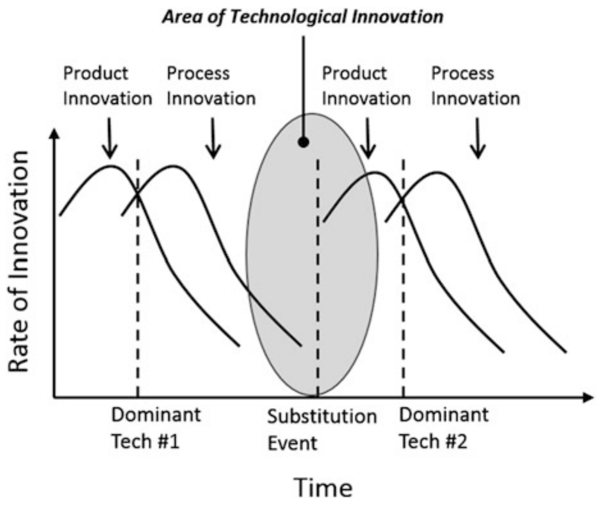


Fig. 60.2 Technology cycles and technological innovation (based on [22])

market, eventually new technology needs to be developed to sustain success. This forces a rapid increase in the rate of innovation, leading to substitute technologies via the technological-innovation process.

In the manufacturing domain, data are being used in new ways that are beginning to enable near-real-time decision-making. Data-driven decision-making is at the core of Industries 4.0, industrial internet of things (IIoT), and smart manufacturing strategies. Significant innovations in data-driven techniques were achieved in the 1980s, but other technological innovations were dominant at that time. As the 2000s approached, the rate of manufacturing-related technical innovation decreased, causing manufacturing to look for new avenues to grow and increase productivity. Manufacturing is again in a time of increased innovation and then we believe the shaded area of technological innovation shown in Fig. 60.2 is imminent. New technologies and new integrations of technologies are revolutionizing the way manufacturing is conducted.

60.2.3 Control Theory Related to Manufacturing Decisions

Control means measuring a quantity or condition in a system and applying a determined quantity or condition to the system to correct or limit the deviation of the measured value from a desired value [23]. Using the word "system" in control problems refers typically to a representation of the actual thing that someone is trying to control.

In engineering, mathematical modeling is a common way of representing a system for controls analysis [23]. Modern control theory has become popular for analyzing complex systems, which often have multiple inputs and outputs as parts of the overall system [23]. A popular method for analyzing these types of complex systems is state-space analysis [24].

In our work, without pretension of being exhaustive, we were less interested in the formulation of representative models. Our interest was in developing a foundational structure to describe the behavior of the system completely at any point in time. That is important for being able to accurately assess the decision-making process. This is why we were interested in control theory – specifically state-space analysis.

While modern control theories provide great values to the engineering domain, they tend to lack complete diagnostics to facilitate controlling the decision-making process. We must also review control in the contexts of management and human factors. Management-control systems include human-resource tools. Organizations might employ management-control techniques in budgets, rules, operating procedures, and performance-appraisal systems to help gain control over employee behaviors [25].

Performance-appraisal systems may include goal setting, which is important to achieving organizational objectives [26]. Organizations implement goal setting with employees because studies show goal setting supports positive motivation and contributes to improved employee performance [27]. Goal setting has also been shown to create competition amongst employees and teams, which increases motivation throughout an organization [28] and improves decision-making processes [27].

Since the 1960s, organizations have used Drucker's [29] work, "Management by Objectives," to control behaviors. Drucker's work has five steps: (1) define organizational objectives, (2) set worker objectives, (3) monitor progress, (4) evaluate performance, and (5) reward results. In the first step, management describes the organization's vision and objectives to the employees. In the second step, each employee meets with management to set specific goals for the employee. The third and fourth steps relate to monitoring and measuring the progress of each employee's goals and providing an evaluation at the end of the performance period. In the last step, the organization rewards each employee based on his/her results.

In Drucker's theory, goal setting is an integral part in all levels of an organization. Ceresia [27] suggested robust management control is supported by both taking into account Drucker's guidance and ensuring positive employee motivation. However, Drucker's theory and Ceresia's recommendations also lack guidance in continuously assessing organization objectives and goals.

Simon [30] published directly on the topic of using control systems to drive strategic renewal. He defines management control systems as "formal, information-based routines and procedures managers use to maintain or alter patterns in organizational activities." Simon also outlines a business strategy with four variables that require assessment. He called these variables "levers of control," which he defined as belief systems, boundary systems, diagnostic-control systems, and interactive-control systems.

We are most interested in the diagnostic-control-systems lever, which provides controls in an optimal spot of the organization because input controls and process standardization do not provide diagnostic management. Input controls maximize creativity but increase risks to cost controls, while organizational goals and standardization minimize creativity and innovation. Diagnostic control systems monitor organizational outcomes, which get compared against important performance dimensions of a strategy. Simon called these "critical performance variables."

In manufacturing industries, critical performance variables are called key performance indicators (KPIs). Simon suggested using KPIs to track the probability of meeting goals or the largest potential for gain over time. These categories of KPIs are considered effectiveness criterion and efficiency criterion, respectively.

The standard ANSI/ISA 95 [31] provides guidance to integrating control systems into enterprise hierarchies. The standard describes a pyramid hierarchy starting with an enterprise level at the top, then moving down to an operations-management level, then a sensing and control level, and finally a devices level. The standard, itself, focuses on the operations-management level.

In Fig. 60.3, we combine the work of Ogata [23], Drucker [29], Simons [30], and ANSI/ISA 95 [31] to form a model for strategy diagnosis in a manufacturing-enterprise-control-system integration. This model demonstrates how organizational strategies, structures, skills, and cultures could evolve according to Tushman [22].

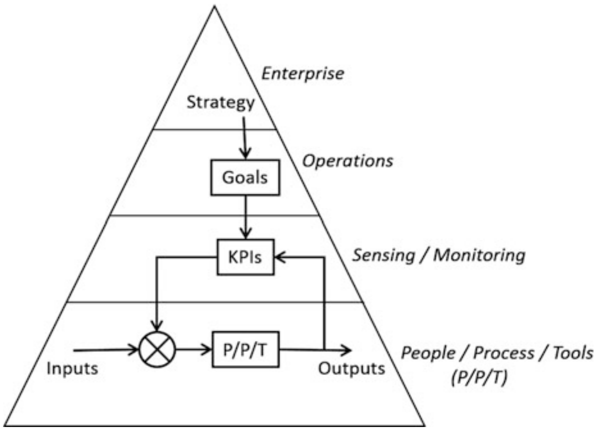


Fig. 60.3 Strategy diagnosis in an enterprise-control-system integration (based on [23, 30, 31])

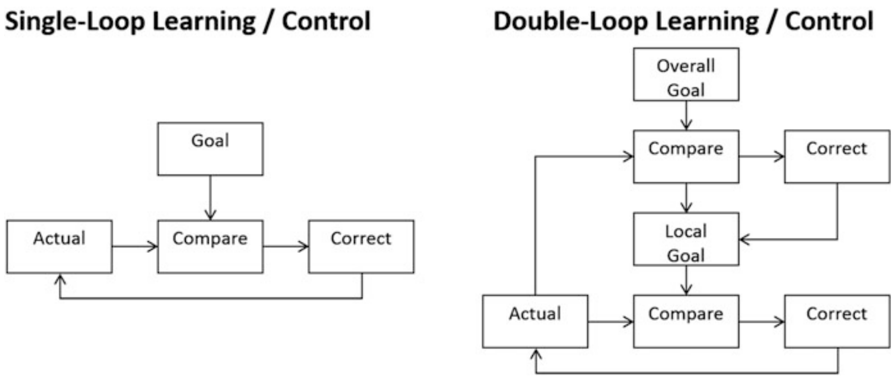


Fig. 60.4 The single-loop learning process compared to the double-loop-learning process (based on [33])

The model depicted in Fig. 60.3 provides a good foundation for controlling the strategies of organizations implementing smart manufacturing, but, like Cooper's [21] model and Drucker's [29] theory, our model for strategy diagnosis may be susceptible to the various types of change, resulting in organizations pursuing strategies that are no longer ideal.

Argyris' [32] developed the concept of double-looping learning. We can represent the concept as a control system. Examples of single-loop-learning and double-loop-learning as control systems are shown in Fig. 60.4. In the double-loop example, there are two "sensors." The first sensor measures the system output in context to the local goal. The second sensor measures the system output in context to the overall goal.

In double-loop learning, the system inputs are modified based on the system output compared to the local goal, but the local goal may also be modified in light of the system output not trending toward the overall goal. The system could also be controlled by modifying the overall goal instead of the local goal.

60.3 Process Definition for Knowledge-Driven Decision-Making in Smart Manufacturing Technology

Informed by literature cited in Sect. 60.2, we developed a concept for controlling decision-making through the technological-innovation process to close the gaps we identified in previous works, specifically continuous assessment of goals and why an organization pursues the technological innovations that they do.

Double-loop learning is a control technique for managing change [32–34]. - Double-loop learning may be used by an organization to decide when to increase the rate of the innovation. Double-loop learning is a way for organizations to break the single-loop learning pattern of “this is the way we have always done it.” Strategically driven organizations typically define governance goals to influence actions that lead to results and consequences. The goals define why organizations do what they do. The actions are what the organizations do. The results and consequences are what organizations obtain.

In single-loop learning, organizations only modify what they do (actions) based on what the organizations obtain (results). This is a process of repeated attempts at the same problem with no variation of method and without ever questioning the goal. With double-loop learning, organizations modify what they do (actions) and reevaluate why they do what they do (goals) basing both on what the organizations obtain (results). This is a process of modifying goals in light of experience or possibly rejecting goals all together after multiple failed attempted [34].

Double-loop learning is important to the technological-innovation process because of the 40–50% failure rate of new-commercialized technology as estimated by Chiesa [16]. While single-loop learning may assist organizations in developing the smart manufacturing technology, double-loop learning would ensure the technology is the right technology the organization should be pursuing.

Double-loop learning, knowledge bases, and popular management-control techniques are integrated into the decision-making process to form our knowledge-driven decisions concept shown in Fig. 60.5. Our concept represents the decision-making process based on generated knowledge and experience. In the decision-making process, our knowledge represents a group of “answers” to previous, and even future, questions. Our decision represents our recognition of a question. We

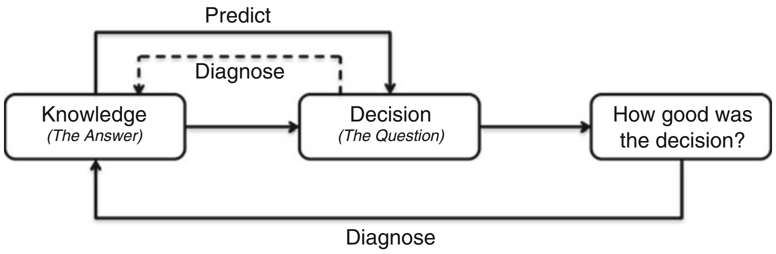


Fig. 60.5 The knowledge-decision cycle for knowledge-driven decision-making

use our knowledge to make decisions. Unfortunately, that is often where the process stops. Due to our interactions and pressures with the work environment, a secondary question of "how good was the decision?" is not often addressed.

For smart manufacturing to be successful, we must analyze each decision outcome against the expected results. This analysis enables the decision maker to diagnose if the knowledge used to make the decision was sufficient or if new knowledge is required to improve the decision in the future. This is a feedback control loop. But just knowledge supports making decisions, we can also use the knowledge to predict the type of decisions we can make in the future. That prediction (i.e., prognosis) activity is a feed-forward control loop that enables secondary diagnosis of the knowledge to determine if new knowledge is needed.

For example, double-loop learning may be integrated with a stage-gate process as shown in Fig. 60.6 by setting an overall goal at the start of the process, defining local goals for each stage of a design or manufacturing process, and then comparing the output of each stage with the overall goal. Requirements are managed within each stage to ensure design, analysis, test, and evaluation, meets the local goals.

In developing our concept, we assume the activities of the discovery, development, and deployment stages are out of scope. We represent each stage as a system, enabling us to integrate each stage using system-of-systems methods. Stage-internal processes and methods are irrelevant; we focus on the inputs and outputs of each stage and the interactions required between each stage.

We also recognize that feedbacks could come from many different disciplines (e.g., marketing, finance, and supply chain). We assume those feedbacks are made directly to the stage that has a need to know. Therefore, those feedbacks are out of scope for this work.

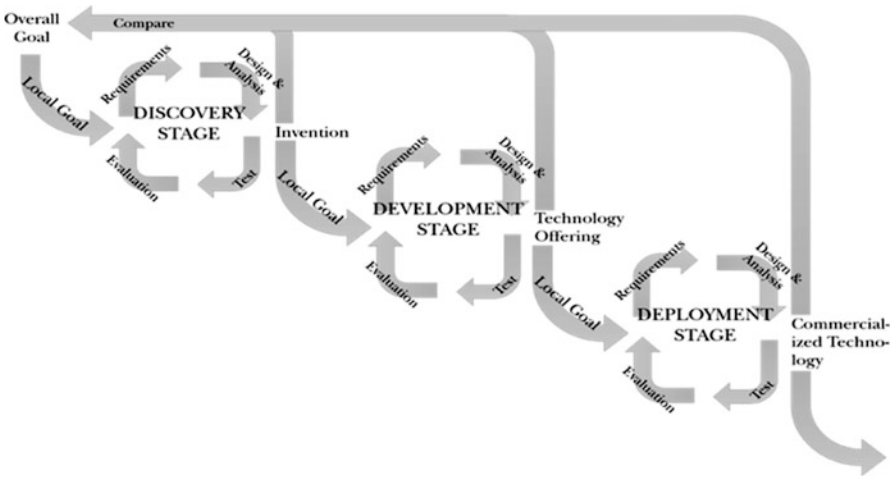


Fig. 60.6 Phase-gate process definition for the management of technological innovation with integrated double-loop control

The first step in employing our concept is to determine the overall goal(s) of the decision cycle – in this case, deploying a new technology. Goal development should follow commonly used processes. We are not proposing a change to existing goal-setting practices; our concept is concerned with making goal assessments a continuous process throughout the decision-making process.

Next, the overall goal(s) are broken down to appropriate localized goals for each stage of the technological-innovation process. This process is similar to developing a work breakdown structure in project management. Local goals should be determined for each stage based on the expected outcome of each stage. In addition, the outputs of each stage become a part of the inputs for the next stage.

When a stage is near completion, an evaluation of the stage's requirements and localized goals is conducted. The results of the evaluation would determine if the technology is ready to move to the next stage. If the technology passes the stage evaluation, that stage is complete. But the technology doesn't move to the next stage yet. The technology needs to be assessed against the overall goal(s). For example, in the discovery stage potential, questions that should be asked are:

- Does the technology align with the overall goals of the organization or vice-versa?
- Would the technology provide continuous value or provide a competitive advantage?
- Are there any other overall goals to which the technology could align?

We adapted Cooper's [21] purpose for gates to our concept. The purpose of each gate is to provide an assessment of the quality of the technology while ensuring the right technologies and overall goals are pursued. Gates are meant to deal with three quality issues in the technological-innovation process [21]: (1) quality of execution in the process, (2) business rationale for the technology and overall goals, (3) and quality of the action plan for controlling the process.

60.4 Discussion

Our example for technology deployment using our decision-making concept could be applied to the concept of the "Digital Thread" and model-based enterprise (MBE). The digital thread is concerned with how data flow between engineering, manufacturing, business processes, and across supply chains. A MBE approach uses these models, rather than documents, as the data source for all engineering activities throughout the product lifecycle. The core MBE tenets are models used to drive all aspects of the product lifecycle, and data are created once and reused by all downstream data consumers.

Hedberg et al. [35] proposed a conceptual lifecycle information framework and technology (LIFT) for digitally integrating all phases of the product lifecycle. Digital thread is key to a successful deployment of the framework and requires an accurate definition of the product (system). The product definition includes the shape, context, and behavior of the system. For example, the shape defines geometry and associated parameter configuration requirements. The context provides information for the various viewpoints across the product lifecycle, such that information is available for each function/role in the lifecycle at the time when the information is needed. Lastly, the behavior describes how the system is required to interact in a given context (e.g., how the system should interact with a cutting tool in manufacturing or how the system should interact with the product end-user).

The LIFT concept is described in three layers: (1) linked-product-lifecycle data, (2) data certification and traceability layer, and (3) data-driven applications. Data are linked together across the entire product lifecycle using agent-based methods. A certification and traceability layer would ensure trust in the linked data by adding meta-data denoting who had done what to the data and when it was done. Lastly, data-driven applications leverage data for knowledge bases, decision support, requirements management, and control.

Combining our decision-making concept with the LIFT concept supports a new paradigm that treats the product lifecycle as a cyber-physical-social system. In this context, computer-aided technologies (CAx), machines, products, and people are combined, analyzed, and measured for performance outputs and decision outcomes. Decisions are made under uncertainty throughout the entire lifecycle. How do we reduce the uncertainty and variation? Further, how do we define and understand stakeholder needs, convert those needs into system requirements, and ensure the system attributes comply with the requirements? Answering those questions relates to the feedback and feed-forward loops in our decision-making concept. Machine learning techniques may be applied to the linked data layer of the LIFT concept. This enables knowledge to be built by diagnosing decisions in the lifecycle. That knowledge is then used to predict the types of decisions that could be made, so before any decision is made, it is possible to determine whether sufficient knowledge exists.

We invested heavily in double-loop feedback during the development of our decision-making concept. Single-loop control has many uses, but our literature review identified several gaps with single-loop control in decision-making activities. The primary gap is the overall goal in development, and manufacturing projects is not being reassessed. We believe the integration of double-loop control in our concept results in better decisions through reducing the risk of bullwhip or oscillatory effects.

While bullwhip effects traditionally apply to distribution channels, the decision-making process can experience oscillatory swings when not controlled properly. Our concept ensures the inputs, outputs, and requirements of decision are carefully assessed at the appropriate point in time against the overall goals of the organization. This ensures a stable and steady flow through the technological-innovation process. Without our concept, deploying smart-manufacturing technologies could potentially experience performance swings above and below optimal states as technology is aligned to the goals of the various process stages in a manufacturing enterprise, the eventual development of new processes, and the overall goals of the organization.

60.5 Conclusion

The successful maturing of a new concept through commercialization requires nurturing and oversight, which only proper management controls would provide, in our case, with the use of double-loop learning. The process of technological innovation and its management must coexist and complement each other through the decision-making process. Cooper [21] outlined eight key factors of success in product development from which we extracted the following technology development insight:

- Quality of executing the technological-innovation process is key to an eventual product's success.
- Quality of executing (e.g., market research, technical assessments, and business analysis) the discovery stage of the technological-innovation process is pivotal to the success of an eventual product.
- Quality of executing marketing activities, including building the voice of the customer, potentially increases the success rate of technology by over 100%.

Our concept for controlling the decision-making process supports Cooper's [21] eight key factors of success. Furthermore, the key factors could be the foundation for assessment criteria throughout the concept. This would enable the application of quality-assurance-like methods to decision-making.

We've begun to show that proper management and technological innovation are critical for successful deployment of smart manufacturing. We've also proposed literature-supported process definitions for technological innovation and a concept for controlling it. We believe organizations would benefit from understanding the relationship of the proposed concept and activities of each phase in technological-innovation process. Additional research is needed in quantifying measures throughout the technological-innovation process.

We recognize that evaluation-based decision outcomes are subjective. Additional research in evaluation criteria and methods for implementing stage-gates into our concept is needed to make outcomes objective. For example, we are interested in how decision trees, heuristics, Markov chain, and Bayesian networks could assist in evaluating the outcome of decisions and how knowledge building could be automated.

Also, additional research is needed to develop measures related to output–input relationships in decision-making. The goal of these measures would be to provide the user of our concept with an efficient way to determine the effectiveness and performance efficiency of each decision against an overall goal. This would help the user determine if an overall goal needs to be revised in light of the work determined by the local goals or vice-versa.

In closing, we believe our concept supports effective and efficient management of the technological-innovation process. Our concept for data-driven decision-making could enable a harmony between the technological-innovation process and its management that would be supported.

Disclaimer and Acknowledgments The work presented in this paper is an official contribution of the National Institute of Standards and Technology (NIST) and not subject to copyright in the United States. Certain commercial systems are identified in this paper. Such identification does not imply recommendation or endorsement by NIST. Nor does it imply that the products identified are necessarily the best available for the purpose.

We wish to thank Mark Carlisle (NIST), Vijay Srinivasan (NIST), and the conference peer-reviewers for their reviews and comments.

References

1. Technology (2015) Entry 4b. In: The Oxford English dictionary, Online edn. Oxford University Press, Oxford
2. Innovation (2015) Entry 1a. In: The Oxford English dictionary, Online edn. Oxford University Press, Oxford
3. Management (2015) Entry 1a. In: The Oxford English dictionary, Online edn. Oxford University Press, Oxford
4. Tidd J, Bessant JR (2009) Managing innovation: integrating technological, market, and organizational change, 4th edn. Wiley, Hoboken
5. Li M, Gao S, Wang CC (2006) Real-time collaborative design with heterogeneous {cad} systems based on neutral modeling commands. J Comput Inf Sci Eng 7:113–125. doi:10.1115/1.2720880
6. Kusiak A (1993) Concurrent engineering: automation, tools, and techniques. Wiley, New York
7. da Silva CES, Salgado EG, Mello CHP, da Silva Oliveira E, Leal F (2014) Integration of computer simulation in design for manufacturing and assembly. Int J Prod Res 52:2851–2866. doi:10.1080/00207543.2013.853887
8. Yang K, El-Haik B (2009) Design for six sigma: a roadmap for product development, 2nd edn. McGraw-Hill, New York
9. Usher JM, Roy U, Parsaei HR (1998) Integrated product and process development: methods, tools, and technologies. Wiley, New York
10. Hedberg TD Jr, Hartman NW, Rosche P, Fischer K (2017) Identified research directions for using manufacturing knowledge earlier in the product life cycle. Int J Prod Res 55:819–827. doi:10.1080/00207543.2016.1213453
11. Knight KE (1967) A descriptive model of the intra-firm innovation process. J Bus 40:478–496. doi:10.2307/2351630
12. Porter ME (1990) The competitive advantage of nations. Free Press, New York
13. Freeman C, Soete L (1997) The economics of industrial innovation, 3rd edn. MIT Press, Cambridge, MA
14. Hollen RMA, Van Den Bosch FAJ, Volberda HW (2013) The role of management innovation in enabling technological process innovation: an inter-organizational perspective. Eur Manag Rev 10:35–50. doi:10.1111/emre.12003
15. Malnight TW (2001) Emerging structural patterns within multinational corporations: toward process-based structures. Acad Manag J 44:1187–1210. doi:10.2307/3069396
16. Chiesa V, Frattini F (2011) Commercializing technological innovation: learning from failures in high-tech markets*. J Prod Innov Manag 28:437–454. doi:10.1111/j.1540-5885.2011.00818.x
17. Schilling MA (2005) Strategic management of technological innovation. McGraw-Hill/Irwin, New York
18. Amabile TM (1996) Creativity and innovation in organizations, Harvard Business Review, 1/5/1996

19. Lewin K, Heider F, Heider GM (1936) Principles of topological psychology, 1st edn. McGraw-Hill book company, inc., New York
20. Hoegl M, Parboteeah KP (2007) Creativity in innovative projects: how teamwork matters. J Eng Technol Manag 24:148–166. doi:10.1016/j.jengtecman.2007.01.008
21. Cooper RG (2001) Winning at new products: accelerating the process from idea to launch, 3rd edn. Perseus Pub, Cambridge, MA
22. Tushman ML, Reilly CAO III (1996) Ambidextrous organizations: managing evolutionary and revolutionary change. Calif Manag Rev 38:8–30
23. Ogata K (2002) Modern control engineering, 4th edn. Prentice Hall, Upper Saddle River
24. Ogata K (1967) State space analysis of control systems. Prentice-Hall, Englewood Cliffs
25. Flamholtz E (1996) Effective organizational control: a framework, applications, and implications. Eur Manag J 14:596–611. doi:10.1016/S0263-2373(96)00056-4
26. Warren K (2008) Strategic management dynamics. Chichester/Hoboken, Wiley
27. Ceresia F (2011) A model of goal dynamics in technology-based organizations. J Eng Technol Manag 28:49–76. doi:10.1016/j.jengtecman.2010.12.004
28. Flamholtz E (1996) Effective management control: theory and practice. Kluwer Academic Publishers, Boston
29. Drucker PF (1954) The practice of management, 1st edn. Harper, New York
30. Simons R (2013) Levers of control: How managers use innovative control systems to drive strategic renewal. Harvard Business Press, Boston
31. The International Society of Automation (2013) Enterprise-control system integration – part 3: activity models of manufacturing operations management. ISA, North Carolina
32. Argyris C, Schön DA (1978) Organizational learning. Addison-Wesley Pub. Co, Reading
33. Reinertsen DG (1997) Managing the design factory: a product developer's toolkit. Free Press, New York
34. Argyris C (2008) Teaching smart people how to learn. Harvard Business Press, Boston
35. Hedberg T Jr, Barnard Feeney A, Helu M, Camelio JA (2016) Towards a lifecycle information framework and technology in manufacturing. J Comput Inf Sci Eng 17(2):021010. doi:10.1115/1.4034132

Chapter 61
Patterns for Modeling Operational Control of Discrete Event Logistics Systems (DELS)

Timothy Sprock

Abstract Designing smart operational control systems for discrete event logistics systems (DELS) requires a standard description of control behaviors executed at the operational management level of DELS control. In this paper, we propose a set of patterns for modeling the operational control mechanisms, organized by classes of control questions that all DELS must be able to answer. The pattern for each control question includes an analysis-agnostic functional definition of the control problem for that question, as well as a mapping of the decision variable in that problem to a particular function and execution mechanism in the base system.

Keywords System design methods • Smart manufacturing • Discrete event logistics systems

61.1 Introduction

Discrete event logistics systems (DELS) are a class of dynamic systems that transform discrete flows through a network of interconnected subsystems [1]. These include systems such as supply chains, manufacturing systems, transportation networks, warehouses, and health care delivery systems. Traditionally, each specialized kind of DELS has been regarded as a distinct class of systems requiring its own dedicated research and development. However, these systems share a common abstraction, i.e., *products* flowing through *processes* being executed by *resources* configured in a *facility* (PPRF), and they appear together in integrated models of the enterprise. For example, production systems might integrate storage and fulfillment capabilities as well as material handling and transportation systems, and supply chains might integrate flows between warehouses, transportation systems, and manufacturing or health care facilities.

The increasing size, integration, and complexity of next-generation smart DELS require more robust engineering design methods. Fundamental to more robust design methodologies are explicit system specifications and more powerful search

T. Sprock (✉)
National Institute of Standards and Technology, Gaithersburg, MD, USA
e-mail: timothy.sprock@nist.gov

A.M. Madni et al. (eds.), *Disciplinary Convergence in Systems Engineering Research*, https://doi.org/10.1007/978-3-319-62217-0_61

and decision support algorithms; see [2] for an example in the warehousing context. Next-generation smart control systems must integrate more information feedback from sensors in the plant and from global information systems, as well as accommodate greater automation. These systems are more software intensive than traditional plant designs, which have focused primarily on hardware selection and configuration. A standard description of operational control would enable development of a uniform interface to decision tools, supported by interoperable, or plug-and-play, analysis tools. However, despite progress on modeling the structure and behavior of DELS, a standard representation of operational control problems for DELS remains a challenge.

This paper presents a set of patterns for modeling the operational control mechanisms for DELS. These patterns include an analysis-agnostic description of each control problem that is used to connect the controller's decision problem to the corresponding actuator function and execution mechanism in the base system. The rest of the paper is organized as follows: Sect. 61.2 provides context for modeling operational control in DELS, Sect. 61.2.1 provides an informal introduction to the control problems, and Sect. 61.2.2 describes a simple form for defining control patterns. Section 61.2.3 uses the pattern form to capture each of the functional control mechanisms and provides a reference architecture for assembling control components. Then Sect. 61.2.4 describes a concrete application of the patterns to specifying a smart manufacturing system. Finally, Sect. 61.3 discusses future directions.

61.2 Modeling Operational Control in DELS

Operational control is the manipulation of flows of tasks and resources through a system in real-time, or near real-time. Each task requires or authorizes a DELS to use its resources to complete a portion of the process plan contained in that task. Operational control consists of multiple mechanisms, each including a function in the controller that prescribes actions to be taken and an actuator in the base system (or plant) that executes the prescribed action. A model of operational control is part of a broader approach to modeling DELS that includes a domain-specific language, a reusable component library, and a reference architecture [3, 4]. This broader model is organized into three layers: the structure layer that captures flow networks, process networks, and relationship networks; the behavior layer that describes each DELS using product, process, resource, and facility (PPRF) concepts; and finally, the control layer, which is the focus of this paper.

A standard analysis-agnostic representation of each control problem is necessary to bridge the gap between the system model and analysis models that support design methods and operational decision-making for the system, including statistical (description), discrete event simulation (prediction), and mathematical programming (optimization) models. This standard representation of operational control is a set of patterns in the DELS reference architecture [5]. These patterns for operational control can describe the control of existing systems, as well as guide design of control for new or modified systems.

61.2.1 The Operational Control Questions

The operational control layer of the DELS reference architecture is based on a set of control problems organized by questions posed by the base system to the controller and corresponding answers that prescribe control actions for the actuators in the base system to execute [4]. Each control question encapsulates a single function of a controller to manipulate the flows of tasks and resources (note that controllers may leverage several functions jointly, i.e., answer several questions together as is the case for scheduling). These control questions are:

1. "Should a task be served?" (*admission*)
2. If so, then "When should the task be serviced?" (*sequencing*)
3. "By which resource?" (*assignment*)
4. "Where (to which DELS for what process) should the task be sent after it completes the required processing at the current DELS?" (*routing*)
5. "When, and to which state, does the state of a resource need to be changed?" (*change in resource capacity or capability*)

Figure 61.1 illustrates the interaction between the base system and controller (control questions and answers), as well as the control execution mechanisms (actuator components in the base system). For more discussion on the functional requirements of the controller itself (how it answers the control questions), see [4, 6].

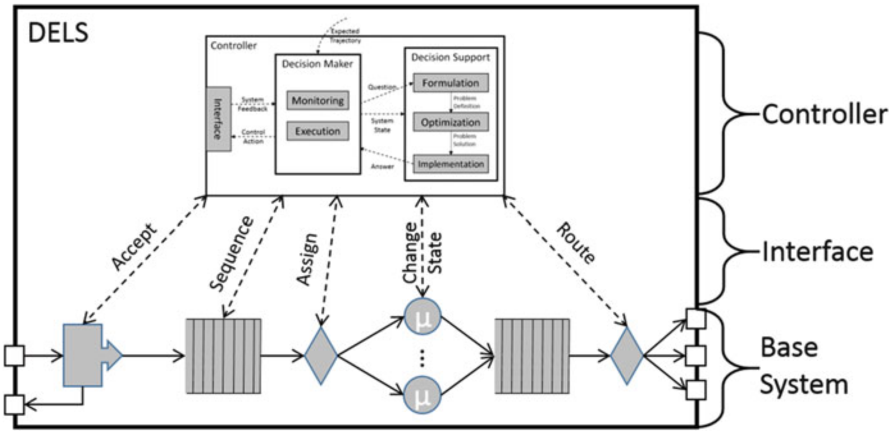


Fig. 61.1 This illustrates the control execution mechanisms (the actuators in the base system), the functional architecture of the controller, and the interaction between the base system and controller (control questions and answers)

For example, production systems use manufacturing resources to service orders or jobs (a kind of task), leading to the following control problems:

1. Admission decides whether or not to accept an order from a customer. The decision may evaluate available production capacity, including raw material inventory on-hand, operator availability, and the current state of resources.
2. Sequencing orders includes decisions such as prioritization (of some customers' orders over others), coordination (of orders outbound to the same customer), batching (similar orders together for efficient processing or transport), delaying (service of an order until a future period or backordering), and splitting (an order into smaller lots to be processed over time).
3. Resource assignment refers to many interrelated problems including assigning scare resources to orders. Manufacturing resources may include labor, critical processing equipment, or material handling equipment. Orders may also require assignment of auxiliary resources such as tools, fixtures, and storage locations to enable process execution.
4. Routing physically or virtually directs orders to resource locations in a facility as required by product's process plan. The routing decision also accommodates unplanned auxiliary processing steps such as exception handling, quality inspection, or unexpected buffer storage, as well as routing optimization for automated guided vehicles (AGVs).
5. Changing the capability or capacity of resources includes replenishment of input material stocks, maintenance on automated systems, changing setups or tooling for machines, or anticipatory movement and prepositioning of inventory or vehicles.

61.2.2 A Form for Defining Operational Control Patterns

Operational control patterns bring together several representations of each control problem to describe the problem in a standard way. A simple form is used in this paper to define the patterns, derived from a function-behavior-structure representation [7, 8]. The components of the form are:

- *Name*: Colloquial identifier of the control problem being addressed (the literature uses various names for the control problems).
- *Question*: Domain-independent, informal "what should I do?" kind of question that the base system poses to the controller (the answer to which is an appropriate control action).
- *Control function*: Transformation, or mapping, of system objects and their state data to actionable control decisions. This transformation formalizes the control question and answer, where the question identifies an applicable control function, and the answers are the control decisions to be executed by the base system. The transformation specifies the functional interface, or signature, of conforming analysis models that answer this particular control question.

- *Decision expression*: A formulation of the control function in terms of one or more binary decision variables (0/1) representing the decision, for use in an optimization analysis model. This part of the pattern describes the control action to be taken when the decision variable has 1 as a value.
- *Actuator function*: Expected effect of the actuator in the base system, for use in simulation models. This is how the control function is carried out by the actuator.
- *Actuator*: Abstract actuator that is capable of carrying out the actuator function. The actuators in this paper are selected from common discrete event simulation modeling components.

The actuator function and actuator itself are used to specify the base system model and corresponding simulation models. The decision variable defines the intent of the optimization model (what question can it answer) that is used to provide decision support. Typically, the actuator and decision variable are developed independently without a shared representation of the control mechanism they are both expressing. The control function provides a common abstraction for these elements to follow, improving interoperability between simulation and optimization analysis models and between the decision support system and the base system it is guiding.

61.2.3 Patterns for Modeling Operational Control of DELS

In Fig. 61.2, the form described in Sect. 61.2.2 is used to specify how to answer each operational control question from Sect. 61.2.1:

1. Admission determines which tasks the controller should admit into the base system. The corresponding decision variable is a yes/no admission choice for each task. This decision is implemented by a function that adds an accepted task to the system's queue. This function is executed by a gate actuator, which is opened, or closed, according to the variable value.
2. Sequencing determines the order, or partial order, that admitted tasks will be serviced by the system. The corresponding decision variables determine the index (position) of each task in the queue. The ordering is implemented by a function that sorts the tasks by an index determined by the decision problem and is executed by the queue where tasks are waiting for service. The task at the head of queue is serviced next.
3. Assignment matches tasks to resources, or partitions the tasks into resource-specific subsets, based on resource capabilities. The decision variable matches tasks to resources, which is implemented by a function that places the task, either virtually or physically, into the assigned resource's queue. Depending on the modeling paradigm, the assignment can be executed by a switch that directs the flow of the task to the resource, which is common in resource-oriented modeling, or it can be executed by seizing the resource from a pool.

4. To route a task for completion of its process plan, the current DELS (or the task itself) must determine where to send the task after the current DELS has completed the requested processes. Specifically, the routing decision identifies the next process required by the task's process plan (*nextProcess*) and selects a suitable DELS to perform that process (*targetDELS*). The actuator function is the composition of two functions: *f* is responsible for evaluating the task's process plan to determine the next required process (*nextProcess*), while *g* is responsible for finding a DELS (*targetDELS*) that is capable of executing the next required process for the task, via, e.g., by directory lookup or call for proposal. These two functions are not necessarily executed in any particular order; i.e., DELS can be solicited to perform each potential process before resolving alternative paths in the process plan or resolve the alternatives and then find suitable DELS. This function is executed by an abstract switch that outputs the task to a particular flow interface, which is connected to the selected target DELS.
5. Deciding to change the capability or capacity of a particular resource uses an abstraction of state to unify the models for answering this control question. The decision variable determines whether to transition resource *m* from state *i* to state *j,* where state is an abstraction for capability or capacity. The pattern for capability states describes the function, process, or service that a discrete state resource can execute at a particular time or a geographic location, i.e., serving tasks at a particular location. The pattern of capacity states describes the amount of work that can be assigned to a particular resource. The functions that change capability and capacity are abstractly modeled as setup (*changeService*) and replenishment behaviors (*increaseCapacity*), respectively. Both behaviors are executed by generating an overhead task, which is accepted, scheduled, and executed by some other resource, such as an operator, maintenance resource, or procurement system.

The abstract actuators corresponding to control questions in Fig. 61.2 are model library components for constructing system models. Figure 61.3 uses these components to formalize the process illustrated in Fig. 61.1. Each component enables the DELS to control the flow of tasks and resources through the system. In Fig. 61.3, a new task enters through the *inTask* port of the DELS and is handled by the *admissionGateway,* a gate type resource that decides whether to admit the task or not. The *admitTask* port (small rectangle on the border of the admissionGateway) interfaces with the controller, which provides the gate with a yes or no (Boolean) decision for each task. The admitted task then flows (filled triangle) into the queue containing the system's *taskSet*. The *sequenceIndex* port interfaces with the controller, which provides the queue with a sequence for the tasks stored in the queue. In this model, resources may be seized from a local *resourceSet* that is owned by the DELS, or may be requested and seized from outside the system through the *ioDELSResource* interface. The *resourceAssignment* port interfaces with the controller, which provides the set of resources that are assigned to serve the task. Once

	Admission	Sequencing
Question	"Should the task be served?"	"When (in what order) should the tasks be served?"
Control Function	$Admit$:Task → $\mathbb{B}$	$Sequence$:Task → $\mathbb{N}$
Decision Expression	$x_i = 1$, if task i is admitted to the system	$x_{ik} = 1$, if task i is serviced k^{th}
Actuator Function	Admit(Task) := {System. TaskSet} ∪ Task	Sequence(TaskSet) := sort(TaskSet, Index) = TaskSet′
Actuator	Abstract Gate	Abstract Queue

	Assignment	Routing
Question	"Which resource should serve the task?"	"Where (to which DELS for what process) should the task be sent after this DELS?"
Control Function	$Assign$: TaskSet × ResourceSet → $\mathbb{B}^{\lvert T\rvert \times \lvert R\rvert}$	$Route$: Task.ProcessPlan → Process × DELS
Decision Expression	$x_{im} = 1$, if task i is assigned to resource m	$x_{OD} = 1$, if the task is output to DELS D for process O
Actuator Function	Assign(Task, Resource) := {Resource. TaskSet} ∪ Task	nextProcess = f(Task.processPlan) targetDELS = g(nextProcess) $Route$(Task) = $g \circ f \vee f \circ g$
Actuator	Abstract Switch or Resource Seize	Abstract Switch

	Change Capability State	Change Capacity State
Question	"Should the capability state of a resource be changed?"	"Should the capacity state of a resource be changed?"
Control Function	$ChangeState$: Resource.State → newState	$ChangeState$: Resource.State → newState
Decision Expression	$\mathrm{X}_{\mathrm{ij}}^{\mathrm{m}} = 1$, if resource m is changed from state i to state j	$\mathrm{X}_{\mathrm{ij}}^{\mathrm{m}} = 1$, if resource m is changed from state i to state j
Actuator Function	$changeService$(Process) := Resource.CurrentService → Process $changeLocation$(Location) := Resource.CurrentLocation → Location	$increaseCapacity$(Quantity) := Resource.CapacityMeasure + Quantity $decreaseCapacity$(Quantity) := Resource.CapacityMeasure − Quantity
Actuator	Abstract `Non-Preemptive Resource Setup' Task	Abstract `Change Resource Pool Size' Task

Fig. 61.2 The patterns for operational control specification in DELS

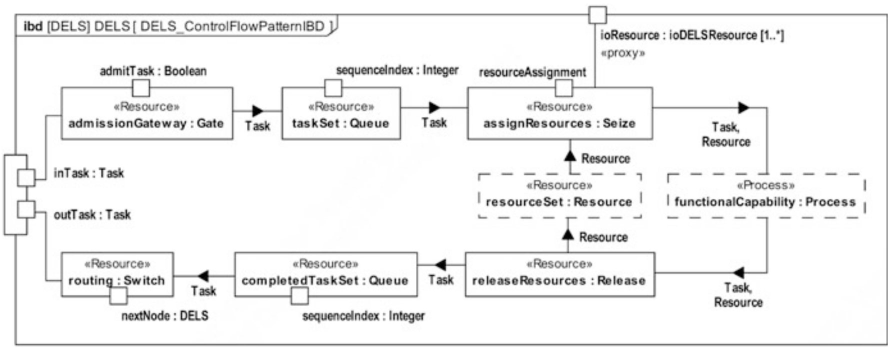


Fig. 61.3 This shows how abstract actuators can be assembled to control the flow of a task through the system

necessary resources have been acquired, the task and resource flow through the remaining internal control processes and actuators as follows:

- The task and resources flow to the *Process* node.
- After processing is complete, the reusable resources are released (*releaseResources)* back to the system's central *resourceSet* or out of the system through the *ioResource* port (flow not depicted).
- The task then enters an outbound queue (*completedTaskSet*) that stores completed tasks waiting to form move batches or the next DELS to be available.
- Finally, the task departs the completed task queue, is routed by the switch (*routing*) to its next DELS, and departs through the appropriate *outTask* port. The *nextNode* port interfaces with controller, which provides the target DELS (which output port) for the task to be routed.

61.2.4 Applying the Patterns: Toward a Smart Manufacturing System Use Case

Designing operational control for DELS includes configuring operational control logic and selecting concrete actuators in the base system that execute the prescribed control actions. The patterns in Sect. 61.2.3 describe operational control problems independently of DELS domains, enabling them to apply to all domains and improve interoperability of base system models with analysis models supporting control decisions, such as optimization. Analysis models are often constructed at a high level of abstraction, but designing a system to execute control decisions requires selecting specific equipment to carry out the function of the abstract actuator (embodiment design). Some examples of concrete actuators for a smart manufacturing system are:

1. The admission gate might be a robotic arm that retrieves the physical workpiece, work in process, and other input resources associated with a task, from an AGV or pneumatic pusher that moves tasks from a centralized conveyor onto the system's local conveyor.
2. Sequencing, and its associated abstract queue for storing tasks waiting for service, might be done by a range of technologies with varying capabilities for executing complex control behaviors. For example, some nonautomated storage solutions might only be capable of simple control behaviors; for example, a gravity-fed conveyor might only be capable of enforcing a first in first out (FIFO) discipline. Some technologies may not be capable of enforcing any sequencing discipline at all; for example, a simple storage rack requires the operator, possibly with the aid of pick lights, to execute the desired sequencing discipline. Simple storage technologies might be augmented with an automated technology, such as a robotic arm capable of picking items from slots, to create a combined system that operates like an automated storage and retrieval systems (ASRS).

3. Assignment of concrete resources to each task depends on whether the task is being brought to the resource or if the resource is being seized and brought to the task. Many systems use both mechanisms. In the case of stationary equipment, such as in a work cell, the assignment mechanism might direct a task into the equipment's queue via pneumatic switch on a conveyor. For resources in a central pool of available discrete units, e.g., input materials, fixtures, or tools, the assignment actuator might be implemented as a robotic arm or AGV that removes the resource from a central buffer and transports it to the work station.
4. Routing tasks from the system often complement admission control and might rely on technologies similar to those that bring tasks into the system. However, in material handling systems where an AGV (or nonautomated worker) delivers the task, the routing behavior must first summon an AGV to the system. Then a robotic arm, or similar mechanism to the admission actuator, can place the task onto the AGV.
5. Change state decisions generate an overhead task for the system, to perform setup or maintenance, reposition a tool or vehicle, order additional inventory, etc. These overhead tasks are assigned, scheduled, and executed by their respective systems, e.g., maintenance, material handling, or procurement, which follow the same control pattern described in Sect. 61.2.3.

61.3 Conclusions and Future Work

To facilitate design and analysis of dynamic, intelligent operational control methods for next-generation DELS, this research identifies a set of control problems, posed as questions to a controller, and patterns for defining the functional, behavioral, and structural aspects of these problems. The patterns use functional descriptions of operational control to connect the base system (plant) model with decision support models, such as optimization and simulation.

The standard description of operational control that is captured by the patterns in this paper supports the development of interoperable, or plug-and-play, analysis tools to answer the control questions. This unifying abstraction of operational control also enables a uniform architecture for each kind of controller, which is especially important for a flexible control architecture and transitioning from traditional centralized, hierarchical control to adaptive, decentralized or holonic architectures [9]. This vision contrasts with each DELS having a different controller architecture depending on its responsibilities, requirements, and the broader control hierarchy selected.

A future goal is formalizing the definitions of operational control for DELS that are documented here informally using patterns. The objective is to formalize the control questions into a canonical set that rigorously partitions the set of all DELS control problems into equivalence classes with a formal equivalence relation (~) based on the functional mapping and associated interface definition. Additional formalization is also required for the mapping between the functional definition and the analysis components.

Future extensions to the control patterns are expected to include concrete system modeling objects as a model library of components, an interface definition for optimization methods, and representative simulation modeling components. Additional reference implementation patterns will be developed that describe integration of simulation components with operational control methods (optimization). The goal is to verify and validate control in simulation and port the logic to a real system, as is common in other engineered systems.

Acknowledgments The author thanks Conrad Bock for his helpful conversations and comments.

Commercial equipment and materials might be identified to adequately specify certain procedures. In no case does such identification imply recommendation or endorsement by the U.S. National Institute of Standards and Technology, nor does it imply that the materials or equipment identified are necessarily the best available for the purpose.

References

1. Mönch L, Lendermann P, McGinnis LF, Schirrmann A (2011) A survey of challenges in modelling and decision-making for discrete event logistics systems. Comput Ind 62(6):557–567
2. McGinnis L, Sprock T (2016) Toward an engineering discipline of warehouse design. 14th International Material Handling Research Colloquium. Karlsruhe, Baden-Württemberg, Germany
3. Thiers G (2014) A model-based systems engineering methodology to make engineering analysis of discrete-event logistics systems more cost-accessible. Ph.D. thesis, Georgia Institute of Technology, Atlanta, GA. https://smartech.gatech.edu/handle/1853/52259
4. Sprock T (2016) A metamodel of operational control for discrete event logistics systems. Georgia Institute of Technology. Ph.D. thesis, Georgia Institute of Technology, Atlanta, GA. https://smartech.gatech.edu/handle/1853/54946
5. Cloutier RJ, Verma D (2007) Applying the concept of patterns to systems architecture. Syst Eng 10(2):138–154
6. Sprock T, McGinnis LF (2015) A conceptual model for operational control in smart manufacturing systems. IFAC PapersOnLine 48(3):1865–1869
7. Gero JS (1990) Design prototypes: a knowledge representation schema for design. AI Mag 11 (4):26
8. Umeda Y, Takeda H, Tomiyama T, Yoshikawa H (1990) Function, behaviour, and structure. Appl Artif Intell Eng V 1:177–194
9. Dilts DM, Boyd NP, Whorms HH (1991) The evolution of control architectures for automated manufacturing systems. J Manuf Syst 10(1):79–93

Chapter 62
Toward Automated Generation of Multimodal Assembly Instructions for Human Operators

Krishnanand N. Kaipa, Carlos W. Morato, Jiashun Liu, and Satyandra K. Gupta

Abstract Factories of the future will be expected to produce increasingly complex products, demonstrate flexibility by rapidly accommodating changes in products or volumes, and remain cost competitive by controlling capital and operational costs. Networked machines with built-in intelligence will become the backbone of these factories. Humans will continue to play a vital role in the operation of the factories of the future to achieve flexibility at low costs. These factories will extensively rely on digital product models to drive the factory operation. Reducing the overall lead-time and compressing the production schedule require that instructions for carrying out these tasks be generated automatically from digital product models. Visual computing is a key technology for rapidly generating instructions for assembly tasks in the factories of the future. Recent advances in modeling systems, physics-based simulations, virtual environments, and visualization techniques enable the possibility of automated generation of multimodal instructions. This paper presents the design of an instruction generation system that can be used to automatically generate instructions for complex assembly operations performed by humans on factory shop floors. The instructions are visual, with a control on the level of detail in order to reduce the cognitive load on the human operator. An automated motion planning subsystem computes a collision-free path for each part from its initial posture in a crowded scene onto its final posture in the current subassembly. Visualization of this computed motion results in generation of 3D animations.

K.N. Kaipa
Department of Mechanical and Aerospace Engineering, Old Dominion University, Norfolk, VA, USA

C.W. Morato
Mechatronics and Sensors Department, US Corporate Research Center, ABB Inc., Raleigh, CT, USA

J. Liu
Department of Mechanical Engineering and Institute for Systems Research, University of Maryland, College Park, MD, USA

S.K. Gupta (✉)
Department of Aerospace and Mechanical Engineering and Center for Advanced Manufacturing, University of Southern California, Los Angeles, CA, USA
e-mail: guptask@usc.edu

A.M. Madni et al. (eds.), *Disciplinary Convergence in Systems Engineering Research*, https://doi.org/10.1007/978-3-319-62217-0_62

Appropriate text and graphical annotations are also generated. The system's ability to automatically translate assembly plans into instructions enables a significant reduction in the time taken to generate instructions and update them in response to design changes.

Keywords Type your keywords here • separated by semicolons

62.1 Introduction

Advanced manufacturing aims to offer a higher level of customization in products at low cost and significantly compressed product development schedules. In the envisioned factories of the future, a network of smart machines will be utilized to achieve high production efficiency and reduce the environmental impact. These envisioned factories of the future will need to meet the following requirements:

- Produce increasingly complex products
- Demonstrate flexibility by rapidly accommodating changes in products or volumes
- Remain cost competitive by controlling capital and operational costs

Realizing complete automation (e.g., not requiring humans in loop) that meets all three above described requirements does not appear to be feasible in the near foreseeable future. The goal of achieving flexibility at low costs simply means that humans will continue to play a vital role in the operation of the factories of the future. Their role will change from doing routine tasks to performing challenging tasks that are difficult to automate. The rest of this article will focus on the role of humans in supporting assembly and maintenance tasks in the factories of the future.

Human and robots have complementary strengths in performing assembly and maintenance tasks. Humans are very good at perception tasks in unstructured environments. For example, they are able to recognize and locate a part from a bin of miscellaneous parts. They are also very good at complex manipulation in tight spaces. In contrast, robots are very good at pick and place operations and highly repeatable. Robots can perform tasks at high speeds and still maintain precision. Robots can also operate for long periods of times with showing signs of fatigue. These complementary strengths can be leveraged using different models of human-robot collaboration to achieve increased levels of productivity [1–3].

We envision that humans and robots will continue to collaborate on assembly and maintenance tasks in the factories of the future. Reducing the overall lead-time and compressing the production schedule require that instructions for carrying out these tasks be generated automatically from digital product models. Visual computing is a key technology for rapidly generating instructions for assembly tasks in the factories of the future. In this paper, we focus on automatically generating instructions for human operators supporting operations in the factories of the future. Computing and networking infrastructure in the factories of future will enable humans to generate and access instructions in the real-time and offer the maximum amount of flexibility in the factory operation. We presented preliminary results in [4].

An assembly sequence is considered as an input to the instruction generation system. The output is a set of multimodal instructions comprising text, graphical annotations, and 3D animations. These instructions will be displayed on a large monitor situated at an appropriate viewing distance from the human carrying out assembly operations on the factory shop floor. An automated motion planning subsystem uses the pertinent data extracted from the assembly plan and computes a collision-free path for each part from its initial posture in a crowded scene onto its final posture in the current subassembly. Visualization of this computed motion in a virtual manufacturing environment results in generation of 3D animations. The system consists of an automated part identification module that enables the human to identify, and pick, the correct part from a set of similar looking parts.

The system's ability to automatically translate assembly plans into instructions enables a significant reduction in the time taken to generate instructions and update them in response to design changes. The multimodal nature of the instructions helps to reduce learning time and eliminates the possibility of assembly errors. The system also accounts for handling of heavy parts and incorporates safety concerns arising due to human-robot interactions. All these features of the instruction generation system significantly contribute to schedule compression. Instruction generation results for a variety of assemblies demonstrate the effectiveness of our approach.

62.2 Related Work

Recent advances in information visualization and human-computer interaction have given rise to different approaches to automated generation of instructions that aid humans in assembly, maintenance, and repair. Heiser et al. [5] derived principles for generating assembly instructions based on insights into how humans perceive the assembly process. They compare the instructions generated by their system with factory provided and hand-designed instructions to show that instruction generation informed by cognitive design principles reduces assembly time significantly. The instructions generated by their automated system were limited to 2D images. Also, the authors restricted their approach to furniture assembly in this work.

Dalal et al. [6] developed a knowledge-based system that generates temporal multimedia presentations. The content included speech, text, and graphics. The authors used a multistage negotiation mechanism to coordinate temporal media. They tested their multimedia generation tool by using it to update patient information to caregivers in hospitals. Zimmerman et al. [7] developed web-based delivery of instructions for inherently 3D construction tasks. The authors used quantitative and qualitative studies to examine factors like user interface, delivery technology and their influence on user interaction level and success in performing inherently

3D operations. They tested the instructions generated by their approach by using them to build paper-based origami models. Kim et al. [8] used recent advances in information visualization to evaluate the effectiveness of visualization techniques for schematic diagrams in maintenance tasks. They focused on diagram highlighting, distortion, and navigation while preserving context between related diagrams.

Instruction presentation systems can benefit from augmented reality techniques. Kalkofen et al. [9] integrated explosion diagrams into augmented reality. The authors developed algorithms to compose visualization images from exploded/non-exploded real-world data and virtual objects. They presented methods to restore missing hidden information in cases where there is a deficiency of information after relocating real-world imagery. The authors showed how to use their approach to automatically compute task-dependent layout and animation of the explosion diagrams.

Henderson and Feiner [10] developed an augmented reality system for mechanics performing maintenance and repair tasks in a field setting. Their prototype supported military mechanics conducting maintenance tasks inside an armored vehicle turret. The system consisted of a tracked head worn display to augment a mechanic's view with text, labels, arrows, and animations. The tasks performed by the mechanics included installation and disassembly of fasteners, lights, and cables, within the cramped turret. The authors carried out a qualitative survey to show that the system enabled easier task handling. Dionne et al. [11] developed a model of automatic instruction delivery to guide humans in virtual 3D environments. The authors proposed a multilevel scheme to address issues like what kind of instructions must be presented to the user in each state and how to generate the final order of instructions.

Brough et al. [12] developed Virtual Training Studio (VTS), a virtual environment-based system that allows (i) training supervisors to create instructions and (ii) trainees to learn assembly operations in a virtual environment. Their system mainly focused on cognitive aspects that enable trainees to recognize parts, remember assembly sequences, and correctly orient the parts during assembly operations. The system allowed three training modes: (a) interactive simulation, (b) 3D animation, and (c) video. The authors presented user test results indicating that the system is able to support wide training preference variety and training for performing assembly operations. A survey of virtual environment-based assembly training applications can be found in [13]. The cognitive aspects of generating instructions that aid the human in correctly recognizing parts bear some similarity to the part identification module in our framework. However, the approach in this paper differs in how the similarity information between the parts is highlighted and presented so that the user picks the correct part.

62.3 System Architecture

Assembly planning usually generates a sequence of high-level assembly tasks. However, these sequences cannot be readily used by robots or human workers as the processes describing how to accomplish each assembly task are not specified in a high-level assembly sequence. Therefore, we address this problem by proposing a low-level assembly planning framework that accepts a high-level assembly sequence as input and generates a set of partially ordered tasks. This output will go through a process of parameter optimization resulting in a linearly ordered assembly sequence with parameters. We have developed an instruction generation system that can automatically translate such linear assembly sequences into multimodal instructions consisting of text, graphical annotations, and animations. Even though our focus in this paper is on instruction generation, we initially provide a brief description of system architecture that drives the instruction generation framework in this section. The overall architecture of our approach is shown in Fig. 62.1. The low-level assembly planning framework consists of five modules.

Task Decomposition This module converts a high-level assembly sequence into a set of partially ordered subtasks. First, a library of state variables are defined to capture conditions on the assembly shop floor (e.g., *At*(Ω,*l*), the set of parts Ω are located at a part storage location ‘; *Secured*(‘), the storage location ‘ is not secured; and *Prepared*(*p*), part *p* is prepared and ready for assembly, etc.) Given the high-level assembly sequence, the state variable definitions library is used to identify the initial and final assembly states. Second, a library of basic subtasks templates that encompass all the assembly operations carried out on the factory shop floor is generated. One example of such a template is shown in Table 62.1. Now, the initial and final assembly states, identified in the first step, are matched with the subtasks template library to enumerate a list of subtasks. This will be followed by binding of variables and elimination of null tasks, giving rise to a partial order on the subtasks that can accomplish the tasks in the given assembly sequence.

Method Selection In this context, we define a method as a prescription of how a subtask must be implemented. For example, if $Transport(p;\ l^{ps};l^{w})$ is a subtask, then the method to transport the part *p* could be manual, by using a trolley, or by using a crane and slings. The weight of the part is used as a parameter to trigger which transport method will be selected. A methods-decision tree is created and used to compute methods for all the partially ordered subtasks.

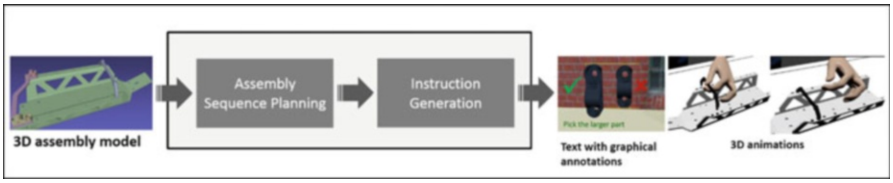


Fig. 62.1 Overall system architecture

Table 62.1 A template of basic subtasks

Basic subtasks	Description
Access(O, l)	Access object *o* at location *l*
Identify(*O*; Ω)	Identify object *o* from object set Ω
Retrieve(*O*)	Retrieve object *o* that was just accessed and identified
Transport(O, $\boldsymbol{l}^1$, $\boldsymbol{l}^2$)	Transport object *o* from location $\boldsymbol{l}^1$ to location $\boldsymbol{l}^2$
Prepare(*O*)	Prepare object *o* for the assembly operation
Clamp(*o*; *l*)	Clamp object *o* to ground at location *l*
Check(*o*)	Check if the object *o* is ready for assembly
Position(O; ϕ; ψ)	Position object *o* at position ϕ and orientation ψ
Insert($\boldsymbol{O}^1$, $\boldsymbol{O}^2$)	Insert object $\boldsymbol{O}^1$, into object $\boldsymbol{O}^2$
Attach($\boldsymbol{O}^1$, $\boldsymbol{O}^2$)	Attach object $\boldsymbol{O}^1$ to object $\boldsymbol{O}^2$
Inspect(*O*)	Inspect object *o* for correctness of assembly
Secure(*l*)	Secure location *l*

Tool Selection The output of the previous module is a set of methods and tools to implement these methods. In the tool selection module, motion planning based on rapidly-exploring random trees (RRT) for coupled 6 DOF tool and simplified human hand models is used to carry out feasibility analysis for each tool identified in the previous module.

Tool Task Decomposition After the tools are selected in this way, tool tasks decomposition will be carried out by using the steps similar to those described in the task decomposition module above. The result is a set of partially ordered tool subtasks, which will be merged with the partial order output from the task decomposition module in order to generate a final partial order on all the subtasks.

Assembly Parameter Selection The motion plan generated by the tool motion planning is quantitative by nature since it is a sequence of 6-tuples that define the motion path. These sequences lose their purpose in the plan. Therefore, the motion plan benefits by adding qualitative information that explains the purpose of movement like how the human worker must lift, move, and operate a tool. Also, assembly parameters like maximum torque to be applied for torque-tools like wrench and screwdriver and maximum force to be applied for force-tools like hammer will be selected in this step.

62.4 Instruction Generation Framework

The input to the instruction generation system is a linearly ordered assembly sequence represented in the extensible markup language (e.g., plan.xml). The contents of each step in the plan.xml file will be used to generate multimodal instructions in the form of text, images, and animations. Next, the various components that constitute the design of the instruction generation system – the language and grammar for text instructions, the process of how the animations will be generated automatically, automated part identification, and instruction presentation – are described briefly.

62.4.1 References and Citation

The language for text instructions will consist of simple verbs such as Identify, Attach, Position, Rotate, Push, Pull, Lift, Lower, Check, Pick, Place, Use, etc. Examples of grammatical constructs for the text instructions include: (1) *Lift* PART? *by* HEIGHT? (2) *Use* HOW MANY? PART? *of type* TYPE? *capacity* CAPACITY?, *length* LENGTH? (3) *Position* PART? *on* LOCATION? (4) *Position* PART? *so that* FEATURE-A? *aligns with* FEATURE-B? (5) *Attach* PART? *at* LOCATION? (6) *Attach* PART-A? *to* PART-B? *so that* FEATURE-A? *mates with* FEATURE-B?

62.4.2 Automated Generation of Animations

The information extracted from plan.xml includes details about the initial scene, labels of the parts and/or subassemblies involved in the assembly operation, and their initial/final postures. This data from each step of the plan is used to invoke a simulation of the assembly operation in a virtual visualization environment, which was developed based on Tundra software. An automated motion planning module interacts with the visualization environment and computes a collision-free motion of a part from its initial posture (e.g., the hood lying in a shelf) to its final posture (e.g., placing of the hood onto the engine compartment of the space frame). Visualization of this computed motion in the visualization environment results in animations that will be appropriately labeled and saved as video clips in a local computer directory. These animations can be augmented with digital human models to generate more realistic animation-based instructions. The MakeHuman and Blender open source computer graphics softwares were used in this paper for this purpose.

We use dynamic multi-random trees (DMRT), a variation of rapidly-exploring random trees, for the purpose of motion planning. In a different paper [14], we report the details of how we combine motion planning and part interaction clusters to automatically generate feasible assembly sequences directly from 3D models of complex assemblies.

62.4.3 Part Identification Instructions

Usually, when a set of parts is presented to a human worker, he/she can easily distinguish among most of them. However, a few may look similar to each other, leading to confusion about which to pick. We have developed a part identification tool to determine automatically the presence of such similar looking parts and present them in a way that allows a worker to identify and pick the correct part. For this purpose, a similarity metric between two parts is constructed based on the following attributes [15, 16]:

1. Part volume and surface area
2. Basic shape statistics such as the types of surfaces and their corresponding areas
3. Gross shape complexity
4. Detailed shape complexity that includes the surface area and curvature information

We consider two assembly examples to illustrate our part identification approach: (a) a simple chassis assembly of five parts, shown in Fig. 62.2a, and (b) a complex chassis assembly of 71 parts, shown in Fig. 62.2b. These two assembly models were obtained from a design team at Vanderbilt University. Table 62.2 shows the dissimilarity matrix for the five parts of the simple chassis assembly, computed using the above technique. The dissimilarity values are in the range [0, 1]. A value of zero means that the two parts are fully similar to each other, and a value of one means that they are fully dissimilar to each other. Now, corresponding to each part $p^{(i)}$, equivalently for each row, the mean dissimilarity $d^{(i)}_{mean}$, with standard deviation values $d^{(i)}_{std}$, is computed over the remaining parts. Any part $p(j)$ whose dissimilarity value lies below $[d^{(i)}_{mean} - d^{(i)}_{std}]$ is considered as similar to $p^{(i)}$. Figure 62.3 shows the values of $d^{(i)}_{mean}$, $d^{(i)}_{mean} + d^{(i)}_{std}$, and $d^{(i)}_{mean} - d^{(i)}_{std}$, for all the five parts. From this figure, note that part $p^{(2)}$ (Radio box 4) is similar to $p^{(3)}$ (Radio box 8). Consider that the human must pick up and assemble $p^{(3)}$ into the current subassembly before picking up $p^{(2)}$. Now, an animation is created, in which the two similar parts detected in the previous phase are lifted vertically and positioned adjacent to each other, with appropriate annotations that enable the human to recognize the correct part for pickup before proceeding for assembly

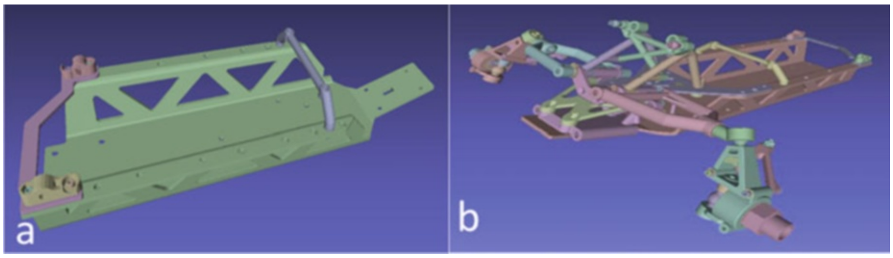


Fig. 62.2 (**a**) A simple chassis assembly of 9 parts. (**b**) A complex chassis assembly of 71 parts

Table 62.2 Dissimilarity matrix for a set of five parts that constitute the simple chassis assembly (0 – part is fully similar; 1 – part is fully dissimilar

Part number	1	2	3	4	5
1	0.000	0.764	0.753	0.667	0.593
2	0.764	0.000	0.312	0.746	0.678
3	0.753	0.312	0.000	0.737	0.656
4	0.667	0.746	0.737	0.000	0.739
5	0.593	0.678	0.656	0.739	0.000

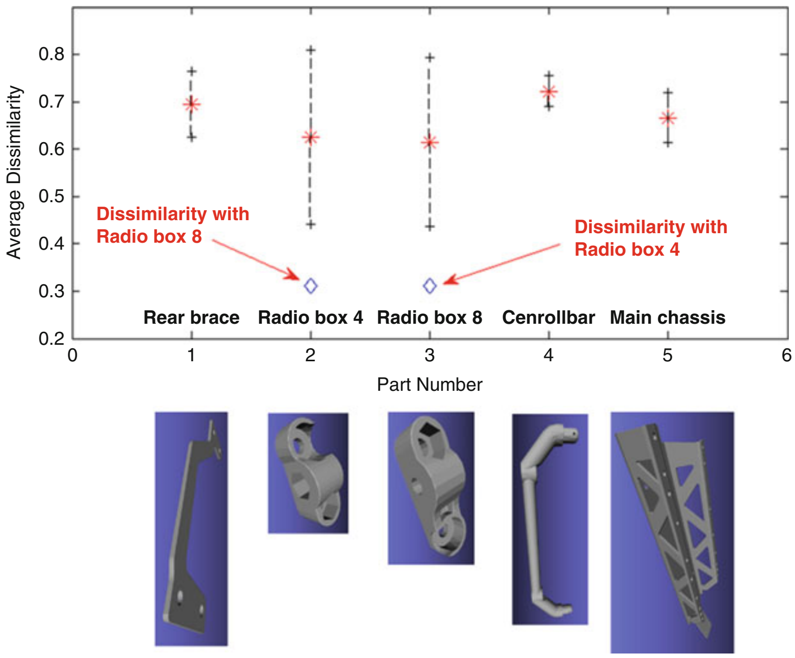


Fig. 62.3 Simple chassis assembly: mean dissimilarity values W.R.T. each part. The mean ± std. values are also shown in the figure

(refer to Instruction 7 in Fig. 62.5). Part identification results for the complex chassis assembly are shown in Fig. 62.4. Note from the figure that the part shown in dotted square is the only part similar to the part p^{45}.

62.4.4 Instruction Presentation

To present instructions to the human worker, a webpage coded with .php scripts is generated by using the data extracted from plan.xml, filling the appropriate language constructs with this data, and querying the corresponding videos stored in the local folders. This framework also allows control on the level of detail in order to reduce the cognitive load on the human operator.

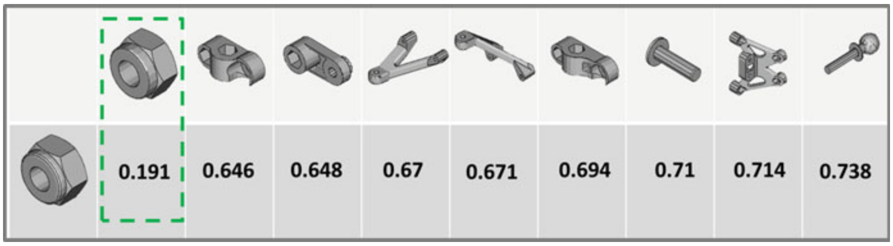


Fig. 62.4 Complex Chassis assembly: dissimilarity values of nine part examples W.R.T. $p^{(45)}$ are shown in the figure. only one part with a dissimilarity value of 0.191 is detected as a similar part

62.5 Results

We report the instruction generation results for the chassis assembly shown in Fig. 62.2a. As mentioned earlier, we have reported our approach to compute feasible assembly sequences directly from a given 3D assembly model in a different paper [10]. Therefore, we consider the following assembly sequence as input to the instruction generation system:

1. Position MAIN CHASSIS at POSTURE 1 on ASSEMBLY TABLE
2. Position CENTER ROLL BAR at POSTURE 2
3. Position REAR BRACE at POSTURE 3
4. Position RADIO BOX 8 at POSTURE 4
5. Position RADIO BOX 4 at POSTURE 5

We assume that the assembly location (Posture 1) is selected by the user. Postures 2–5 are computed by combining information about the posture 1 and the relative/absolute reference frames extracted from the assembly model. Figure 62.5 shows snapshots of the instructions – text, graphical annotations, and 3D animations – generated by the system, for each assembly step. Note from the figure that a part identification instruction precedes every assembly operation, in which a new part must be picked up and assembled onto the current subassembly. This results in a set of ten instructions. Snapshots from the live video footage of a human viewing an assembly instruction and implementing the viewed instruction are shown in Fig. 62.6.

62.6 Conclusions

We presented a design framework to automatically generate instructions for human operators supporting operations in the factories of the future. The system's ability to automatically translate assembly plans into instructions enables a significant reduction in the time taken to generate instructions and update them in response to design

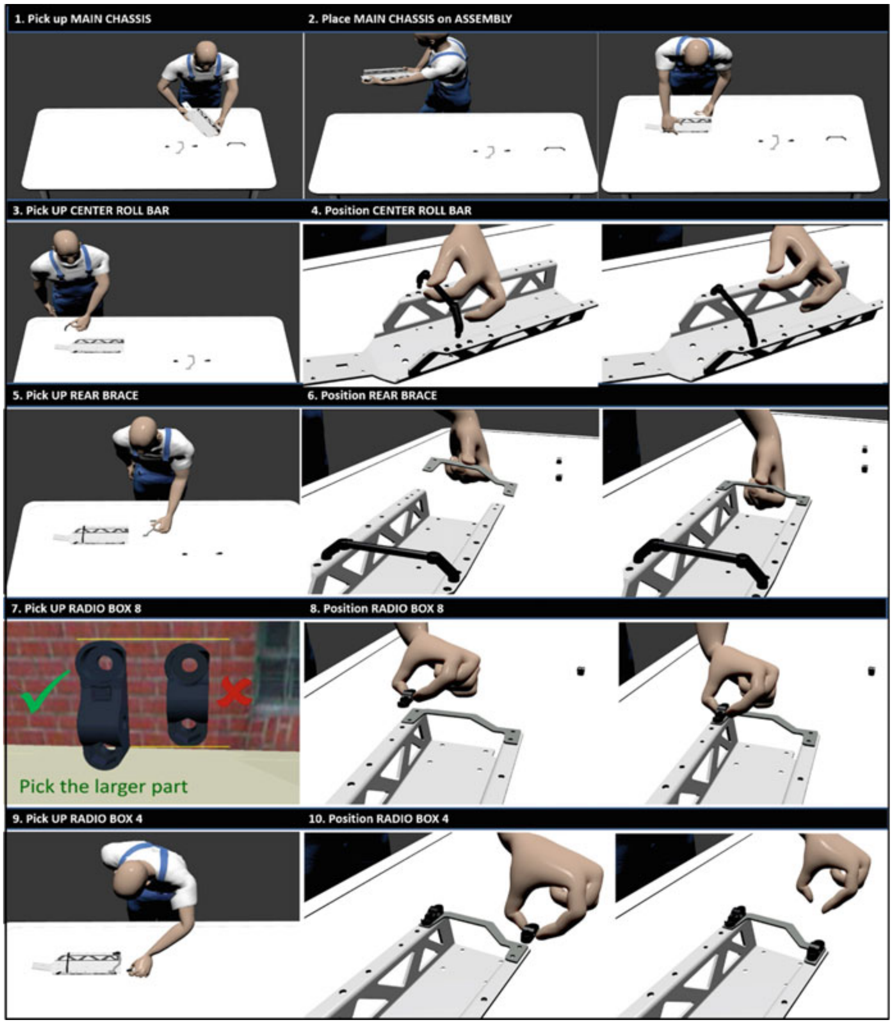


Fig. 62.5 Generation of instructions for chassis assembly

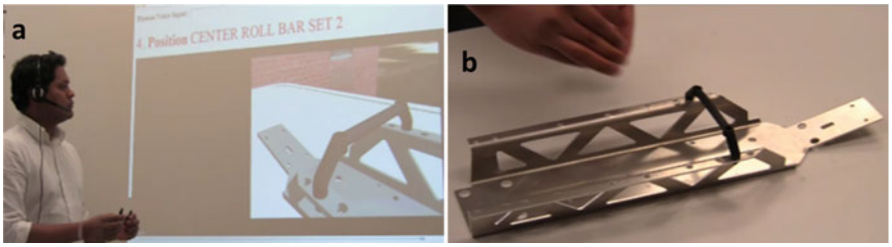


Fig. 62.6 Human viewing instruction and performing assembly

changes. The visual computing methods described in the paper will enable humans to generate and access instructions in the real-time and offer the maximum amount of flexibility in the factory operation. In the current design, motion planning is applied only to parts, and the resulting animations are manually augmented with digital human models. This issue can be addressed by incorporating human models, with increasingly complex degrees of freedom, into the motion planning framework. This results in physically realistic animations that show how to lift and move the parts/tools during assembly while satisfying human motion constraints. In our previous work, we developed other techniques including singulation planning [17, 18], fine-positioning [19], ontology for task-partitioning ontology in human-robot collaboration for kitting operations [20], ensuring human safety in hybrid cells [21], and resolving occlusions during bin-picking [22]. Future work consists of investigating how to integrate them and methods presented in this paper to realize hybrid work cells where humans and robots collaborate to carry out industrial tasks.

Acknowledgments This work was supported by DARPA AVM iFab Project (Contract # FA8650-11-C-7128). This project was a large collaborative effort involving Carnegie Mellon University, Lockheed Martin, Pratt and Miller, University of Maryland, and University of Michigan. We acknowledge and thank all members of our team for their valuable input. We especially would like to thank David Bourne from Carnegie Mellon University for sharing his valuable insights and feedback. We are also thankful to META team at Vanderbilt University for providing assemblies used in this paper.

References

1. Kaipa KN, Kankanhalli-Nagendra AS, Kumbla NB, Shriyam S, Thevendria-Karthic SS, Marvel JA, Gupta SK (2016) Addressing perception uncertainty induced failure modes in robotic bin-picking. Robot Comput Integr Manuf 42(1):17–38
2. Kaipa KN, Kankanhalli-Nagendra AS, Kumbla NB, Shriyam S, Thevendria-Karthic SS, Marvel JA, Gupta SK (2016) Enhancing robotic unstructured bin-picking performance by enabling remote human interventions in challenging perception scenarios. In Proceedings of IEEE International Conference on Automation Science and Engineering, Fort Worth, August 21–24, 2016
3. Kaipa KN, Thevendria-Karthic SS, Shriyam S, Kabir AM, Langsfeld JD, Gupta SK (2015) Resolving automated perception system failures in bin-picking tasks using assistance from remote human operators, In Proceedings of IEEE International Conference on Automation Science and Engineering, Gothenburg, Sweden, August 24–28, 2015
4. Kaipa KN, Morato C, Zhao B, Gupta SK (2012) Instruction generation for assembly operations performed by humans. In Proceedings of ASME 2012 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, pp 1121–1130
5. Heiser J, Phan D, Agrawala M, Tversky B, Hanrahan P (1995) Identification and validation of cognitive design principles for automated generation of assembly instructions. In Proceedings of the Working Conference on Advanced Visual Interfaces, Gallipoli, Italy, pp 311–319
6. Dalal M, Feiner S, McKeown K, Pan S, Zhou M, H*o*llerer T, Shaw J, Feng Y, Fromer J (1996) Negotiation for automated generation of temporal multimedia presentations. In Proceedings of the fourth ACM International Conference on Multimedia, Boston, MA, pp 55–64

7. Zimmerman G, Barnes J, Leventhal L (2003) A comparison of the usability and effectiveness of web-based delivery of instructions for inherently-3D construction tasks on handheld and desktop computers. In Proceedings of International Conference on 3D Web technology, Saint Malo, France, pp 49–54
8. Kim S, Woo I, Maciejewski R, Ebert DS, Ropp TD, Thomas K (2010) Evaluating the effectiveness of visualization techniques for schematic diagrams in maintenance tasks. In Proceedings of the 7th Symposium on Applied Perception in Graphics and Visualization, Los Angeles, CA, pp 33–40
9. Kalkofen D, Tatzgern M, Schmalstieg D (2009) Explosion diagrams in augmented reality. In Proceedings of IEEE International Conference on Virtual Reality, pp 71–78
10. Henderson S, Feiner S (2011) Exploring the benefits of augmented reality documentation for maintenance and repair. IEEE Trans Vis Comput Graph 17(10):1355–1368
11. Dionne D, de la Puente S, León C, Hervás R, Gervás P (2010) A model for human readable instruction generation using level-based discourse planning and dynamic inference of attributes disambiguation. In Proceedings of European Workshop on Natural Language Generation, Athens, Greece
12. Brough JE, Schwartz M, Gupta SK, Anand DK, Kavetsky R, Pettersen R (2007) Towards development of a virtual environment-based training system for mechanical assembly operations. Virtual Reality 11(4):189–206
13. Gupta SK, Anand DK, Brough JE, Schwartz M, Kavetsky R (2008) Training in virtual environments: a safe, cost-effective, and engaging approach to training. CALCE EPSC Press, College Park
14. Morato C, Kaipa KN, Gupta SK (2013) Improving assembly precedence constraint generation by utilizing motion planning and part interaction clusters. Comput Aided Des 45 (11):1349–1364
15. Cardone A, Gupta SK, Karnik M (2003) A survey of shape similarity assessment algorithms for product design and manufacturing applications. J Comput Inf Sci Eng 3(2):109–118
16. Cardone A, Gupta SK (2006) Shape similarity assessment based on face alignment using attributed applied vectors. In Proceedings of CAD Conference, Phuket Island
17. Kaipa KN, Shriyam S, Kumbla NB, Gupta SK (2015) Automated plan generation for robotic singulation from mixed bins. In IROS Workshop on Task Planning for Intelligent Robots in Service and Manufacturing, Hamburg, Germany
18. Kumbla NB, Thakar S, Kaipa KN, Marvel JA, Gupta SK. (2017) Simulation based on-line evaluation of singulation plans to handle perception uncertainty in robotic bin picking. In Proceedings of ASME Manufacturing Science and Engineering Conference, Los Angeles, CA, June 4–8, 2017
19. Kaipa KN, Kumbla NB, Gupta SK (2015) Characterizing performance of sensorless fine positioning moves in the presence of grasping position uncertainty. In Proceedings of IROS Workshop on Task Planning for Intelligent Robots in Service and Manufacturing, Hamburg, Germany
20. Banerjee AG, Barnes A, Kaipa K, Liu J, Shriyam S, Shah N, Gupta SK (2015) An ontology to enable optimized task partitioning inhuman-robot collaboration for warehouse kitting operations. In Proceedings of.SPIE, Next-Generation Robotics II; and Machine Intelligence and Bio-inspired Computation: Theory and Applications IX, 94940H, Baltimore, MD
21. Morato C, Kaipa KN, Zhao B, Gupta SK (2014) Toward safe human robot collaboration by using multiple kinects based real-time human tracking. J Comput Inf Sci Eng 14(1):011006
22. Kaipa KN, Shriyam S, Kumbla NB, Gupta SK (2016) Resolving occlusions through simple extraction motions in robotic bin-picking. ASME Manufacturing Science and Engineering Conference, Blacksburg, June 27–July 1, 2016

Part IX
Systems Engineering Applications

Chapter 63
A Game Theory Perspective on Requirement-Based Engineering Design

Soodabeh Yazdani, Yen-Ting Lin, Wenbo Cai, and Edward Huang

Abstract In the design of large-scale systems, component/subsystems are oftentimes designed by different design teams who utilize shared resources. Requirement-based engineering design is a common practice that allocates the shared resource to each design team and requires each component/subsystems to be designed within that resource limitation. This paper explores limitations and challenges of requirement-based engineering design. Specifically, we use noncooperative game theory to demonstrate that requirement-based engineering design results in suboptimal performance for the entire design project.

Keywords Requirement-based engineering design • Noncooperative game theory

63.1 Introduction

With the development of technology and market needs, design of large-scale systems, such as satellites and aircraft, becomes increasingly complex. These systems are oftentimes too complex to be designed by a single engineer or team. Instead, they are usually hierarchically decomposed into subsystems that are designed separately by various subcontractors or design teams. In doing so, system-level requirements (such as constraint on weight, size, power, length, etc.) are decomposed to subsystem-level requirements so that they can be completely addressed by each design team. This design approach is also known as requirement-

S. Yazdani (✉) • E. Huang
Systems Engineering & Operations Research Department, George Mason University, Fairfax, VA 22030, USA
e-mail: Syazdan2@gmu.edu; chuang10@gmu.edu

Y.-T. Lin
School of Business Administration, University of San Diego, San Diego, CA 92110, USA
e-mail: linyt@sandiego.edu

W. Cai
Department of Mechanical & Industrial engineering, New Jersey Institute of Technology, University Heights Newark, Newark, NJ 07102, USA
e-mail: wenbo.cai@njit.edu

A.M. Madni et al. (eds.), *Disciplinary Convergence in Systems Engineering Research*, https://doi.org/10.1007/978-3-319-62217-0_63

based engineering design. In such an approach, the system-level requirements (such as total weight of a satellite) can be seen as recourses shared by its subsystems. Apparently, the performance and timeliness of the entire system critically hinges on the efficient allocation of those resources.

There are two common approaches to allocate resources (e.g., allowable weight, size, power, length, etc.) in requirement-based engineering design: (1) third party-driven approach and (2) individual-driven approach (Randii R. Wessen, 1998). In third party-driven approach, resource allocation is carried out by a third party committee, whereas in an individual-driven approach the resource allocation is carried out by a project member. However, both of the approaches share some common shortcomings. For example, consider a large system such as aircraft in which an official, namely project director, assigns recourses such as maximum mass to each subsystem. For simplicity, suppose the system is composed of two sub-systems who aims to maximize its power and the system to be designed must weight no more than 400,000 pounds. The project director decomposes the system-level requirements and allocate weight limit of 200,000 pounds to each teams 1 and 2. Team 1 has two design options while team 2 has only one. However, the available options are only known by each design team but not the project director. Option 1 is the only feasible choice for team 1, despite that option 2 allows a better system performance. Table 63.1 demonstrates the potential limits of allocation resources in requirement-based engineering design.

As shown in Table 63.1, under requirement-based engineering design, both teams can only choose option 1, and the performance of the whole system is suboptimal. The entire system could have performed better if teams 1 and 2 are allocated 220,000 and 180,000 pounds, respectively. The purpose of this paper is to analytically demonstrate this inefficiency in requirement-based engineering design.

Therefore, in requirement-based engineering design, design teams focus solely on meeting their own requirement specifications, and there is no incentive to further improve the product. Second, poor decomposition of system-level requirements often limits design alternatives. As a result, the design outcome may not be optimal. While this suboptimal outcome is intuitive, there is a lack of theoretical evidence to illustrate this inefficiency in requirement-based engineering design. This paper intends to fill in that gap. Specifically, we apply noncooperative game theory to show that requirement-based engineering design leads to a suboptimal design outcome.

Table 63.1 Example of resource allocation limits of requirement-based engineering design

Design team	Team 1		Team 2
Requirement	Weight: 200,000 pounds		Weight: 200,000 pounds
Options	Option 1	Option 2	Option 1
Expected performance	Weight: 200,000 pounds Power: 2000 horsepower	Weight: 220,000 pounds Power: 2500 horsepower	Weight: 180,000 pounds Power: 1500 horsepower

Game theory is originally developed in mathematics and economics. It is nowadays widely applied in marketing [22, 23], supply chain [21, 24], and operation management [28] literatures to characterize competitive behavior in market places. It offers effective characterization of interaction between decision makers. It is appropriate to analyze challenges of requirement-based engineering design due to following characteristics:

- Distinct objectives: There are multiple decision makers. Each of them has his/her own objectives, which may be in conflict with each other. This misalignment of objectives also causes potential inefficiency for the entire engineering design.
- Interdependent decisions: Decision of each design team may affect performance/interest of another design team. Thus, a decision maker needs to also consider how others may response to his/her actions.

Hence, there may not be an optimal solution that pleases every decision maker because they compete for their own interests, making traditional optimization inadequate to analyze the decisions. On the other hand, noncooperative game theory uses equilibrium concepts to provide a rational prediction of decision outcomes. Specifically, decisions reach equilibrium when no decision maker can find any better decision without altering the decision of another player.

This paper is organized as follows. Section 2 offers review of literatures. In Sect. 3, we introduce the game theoretic framework and provide analytical results. Section 4 presents a summary and some conclusions and future work we can draw from this research.

63.2 Literature Review

The challenges in implementing requirement-based engineering design have received much attention. Firesmith [11] summarized some practical problems of using requirement-based engineering design which includes poor or ambiguous requirement, incomplete, inconsistent, incorrect and out-of-date requirements, changing requirements over time, etc. For each of these problems, the author suggested some industry best practices. Similarly, Shah et al. [12] also provide extensive review of major challenge. Sabaliauskaite et al. [13] conducted an interview in a large-scale industry domain. They confirmed the results of [11, 12] and indicate that poor communication and cooperation between different design teams is a major cause of problems. Past literature also shows that performance of requirement-based engineering design can be improved by increasing the quality of requirements and more efficient allocation of requirements [18–20].

The limitations of requirement-based design are addressed by [2–10, 14–17]. All of these studies focus on implementation challenges of requirement-based engineering design. Most researchers used either case studies [11, 12, 14–16] or qualitative research interviews [10, 13] to point out the major implementation challenges. In contrast, we apply game theory for a theoretical analysis on the performance of requirement-based engineering design. In other words, we consider a theoretical world where implementation problems do not exist and examine when requirement-based engineering design delivers the optimal design outcome. Specifically, we first consider the common practice where requirement allocation and subsystem designs are carried out by different decision makers. Next, we consider an ideal situation where all of the decisions are made by a central decision maker who aims to maximize the performance of the entire design project. Finally, we compare the performance of the entire project between these two cases to evaluate efficiency of requirement-based engineering design.

63.3 Model and Analysis

In this section, we formulate the engineering design problem and discuss analytical results. We first consider a decentralized scenario where each decision maker makes his/her own decision to maximize his/her own interest. Then we describe another centralized scenario where all of the decisions are jointly made by a hypothetical central planner whose goal is to maximize the interest of the entire system. Finally, we contrast the outcome of these two scenarios to unveil the performance consequence of decentralized versus centralized decision making.

Consider a project director who outsources design of subsystems, or components, to$n$ design teams, each with a single decision maker that we refer to as managers. Each manager i has a reservation value of zero, and he receives payoff R_i from the project director if the designed component meets the requirement. Each component designed by team **i** has **j** attributes, indexed by $j = 1, 2, \ldots, J$, such as weight, length, size, and capacity. Without loss of generality, we assume that smaller attribute values are preferred over large ones. Each team is given by the project manager an allocation A_{ij}. Hence, A_{ij} represents the maximum value of attribute j that the design outcome of team i can take. Each design team i determines the improvement e_{ij} on attribute j at a cost $C_{ij}(e_i)$, where $e_i = [e_{i0}, \ldots, e_{iJ}]$ and C_{ij} increases in e_{ij}. Thus, for team i the final outcome for attribute j is:

$$V_{ij} = a_{ij} - C_{ij}.e_{ij}$$

where V_{ij} is the design outcome for attribute j, a_{ij} is the attribute value without any improvement effort, and e_{ij} shows the amount of improvement design team i exerts. The improvement e_{ij} incurs the following cost C_{ij} to team i:

$$C_{ij}\left(e_{ij}\right) = \frac{1}{2}k_{ij}.e_{ij}^2$$

We assume improvement costs to be convex in e_{ij} so the marginal reduction in the attribute value decreases in effort, and k_{ij} captures the convexity of the cost.

We assume a quadratic function for tractability, and this is a common assumption [25–28].

In requirement-based design, the project director allocates attribute values to each design team hence each design team is allocated an upper limit A_{ij} for each attribute. We assume that $A_{ij} \leq a_{ij}$ and therefore teams need to exert improvement effort to meet their requirements.

The director first determines payments R_i to each team when its design outcome meets the requirements A_{ij}. Next, each team determines their effort e_{ij}. Thus, the director optimization problem is as follows:

$$(\text{P1}) \quad C^{\text{R}} = \arg\min_{\text{R}_i} \sum_{i=1}^{n} \text{R}_i$$

s.t.

$$\pi_i = \arg\max_{e_{ij}} \text{R}_i - \sum_{j=1}^{j} C_{ij}\left(e_{ij}\right) \qquad \text{for all } i \tag{63.1}$$

$$\pi_i \geq 0 \tag{63.2}$$

$$A_{ij} \geq V_{ij} = a_{ij} - C_{ij}e_{ij} \tag{63.3}$$

In problem P1, the project director aims to minimize his total payments to design teams such that the requirements are met. Constraints (63.1) and (63.2) show that each team aims to maximize their payoffs, and constraint (63.3) ensures that the entire project meets its requirement on each attribute.

On the other hand, in a centralized system, the best performance can be derived by solving the following problem:

$$(\text{P2}) \quad C^{\text{p}} = \arg\min_{e_{ij}} \sum_{j=1}^{J} \sum_{i=1}^{n} C_{ij}e_{ij}$$

s.t.

$$\sum_{i=1}^{n} A_{ij} \geq \sum_{i=1}^{n} V_{ij} \quad \text{for all } j$$

In problem P2, we assume that the decisions of each team are made by a central planner whose goal is to meet all of the attribute requirements at a lowest cost. We solve each of the problems and compare their total cost in the following Theorem:

Theorem

1. The total cost for the entire system is higher under requirement-based design

$$C^{\mathrm{P}} \leq C^{\mathrm{R}} \tag{63.4}$$

2. The equilibrium effort level for each team i under requirement-based design is

$$e_{ij}^{*} = \frac{a_{ij} - A_{ij}}{C_{ij}} \text{ for all } j \tag{63.5}$$

3. The equilibrium effort level for each team i in a centralized system is

$$e_{ij}^{*} = \left(\frac{C_{ij}}{k_{ij}}\right)\left(\frac{\sum_{i=1}^{n}\left(a_{ij} - A_{ij}\right)}{\sum_{i=1}^{n}\frac{C_{ij}^{2}}{k_{ij}}}\right) \text{ for all } j \tag{63.6}$$

Proof See Appendix A

This theorem leads to the following managerial insights:

- Equation (63.4) states that in requirement-based engineering design, the subsystems' functional requirements are satisfied but the total cost is higher than when all of the decisions are made centrally. This happens because design teams only meet their given specified requirements which are determined inefficiently in the first place. Consequently, the entire system suffers from higher cost. In other words, the performance of requirement-based engineering design critically depends on the initial allocation of resource (i.e., specification given to each design team).
- From Eq. (63.5) we can see that the optimal amount of improvement for each team i does not relate to cost coefficients. Design teams focus only on meeting the specifications and they have no incentive to improve beyond the specification. But in centralized decision making, the central decision maker allocates the resource to the most efficient way and efforts are exerted accordingly to minimize system-wide cost. Therefore, as Eq. (63.6) shows, the central planner takes into account the cost coefficient in determining the optimal efforts.
- Also Eq. (63.5) says that the optimal amount of improvement depends on the differences between initial attribute and allocated values for each design team i, while based on (63.6), in centralized system the optimum amount of improvement simply depends on the sum of the differences between initial attribute and allocated values for all design teams. Because in decentralized structure, each decision maker concerns only their self-interest with no emphasis on quality or better design. On the contrary, centralized approach can promote design teams to increase their total effort to improve efficiency.

63.4 Conclusion

This paper is an attempt to study limitations of requirement-based engineering design with game theory perspective.

Although the relevant studies have solved some limitations and facilitate some solutions to overcome the requirement-based design issues, the significant challenges still remain unattended, and there is a lack of theoretical justification in assessing the system-wide performance of requirement-based engineering design. Our research provides a new insight on the effectiveness of requirement-based engineering design. For this purpose, we applied noncooperative game theory to model the requirement-based engineering design process. It points out significant results which show that we can't always get the system-wide best performance in the requirement-based engineering design and total performance is usually suboptimal.

In the future, we are looking forward to find an approach toward creating the optimal value of the designed systems by successful implementation of value-driven design [29–31]. Major benefits of value-driven design to the engineering design of complex systems are: (1) allowing design optimization for the entire system during early design phases and for each subsystem/component during later design, (2) preventing design trade conflicts, and (3) avoiding the cost increase and performance decrease accrued by eliminating requirements for attributes at the subsystem/component level [30]. Therefore, one direction for further research is to apply noncooperative game theory to value-driven design and provide scientific insight on how to implement this approach to achieve a better design quality at a lower cost.

Proof of Theorem

This appendix provides the mathematical proofs of the theoretical result in our paper.

Proof of part (4)

In (P1) apparently the director's optimal strategy is lowering in R_i until $\pi_i \geq 0$ is binding. So the problem can be written as:

$$\bar{C}^{R} = \sum_{i=1}^{n} \sum_{j=1}^{J} C_{ij}\left(e_{ij}\right)$$

$$s.t. \qquad e_{ij} \in \arg\min_{e_{ij}} \sum_{j=1}^{J} C_{ij}\left(e_{ij}\right)$$

$$A_{ij} \geq V_{ij}$$

$$A_j = \sum_{i=1}^{n} A_{ij} \;\; \geq \;\; \sum_{i}^{n} V_{ij} = V_j$$

Then we have:

$$C^{\mathrm{R}} \geq \bar{C}^{\mathrm{R}} = \arg\min_{e_{ij}} \sum_{i=1}^{n} \sum_{j=1}^{J} C_{ij}(e_{ij}) \geq C^{P} = \arg\min_{e_{ij}} \sum_{i=1}^{n} \sum_{j=1}^{J} C_{ij}$$

$$s.t. \quad A_{ij} \geq V_{ij} \qquad s.t. \quad A_j \geq V_j$$

$$A_j \geq V_j$$

Proof of part (5)

Given that $A_{ij} \leq a_{ij}$ and profit function of team i decreases in e_{ij}, each design teams optimal decision is to exert minimum effort such that $A_{ij} = a_{ij}$, which this leads to desired result.

Proof of part (6)

To solve problem (P2), consider the lagrangian transformation of the problem:

$$L^{\mathrm{P}} = \sum_{j}^{J} \sum_{i}^{n} C_{ij}(e_{ij}) + \sum_{j}^{J} \left(\sum_{i}^{n} \left(A_{ij} - a_{ij} + C_{ij}(e_{ij}) \right) \right) \qquad \text{for all } j$$

By differentiation with respect to e_{ij}, we get:

$$\frac{\partial L^{\mathrm{P}}}{\partial e_{ij}} = -k_{ij} e_{ij} + \lambda_j \;\, C_{ij} = 0$$

This implies:

$$e_{ij}^{*} = \frac{\lambda_i C_{ij}}{k_{ij.}}$$

Since in optimality

$$\sum_{i}^{n} A_{ij} = \sum_{i}^{n} V_{ij}$$

Therefore:

$$\sum_{i}^{n} A_{ij} - \sum_{i}^{n} V_{ij} = \sum_{i}^{n} \left(A_{ij} - a_{ij} + C_{ij} \left(\frac{\lambda_j \ C_{ij}}{k_{ij}} \right) \right)$$
$$= \sum_{i}^{n} \left(A_{ij} - a_{ij} \right) + \lambda_j \ \sum_{i}^{n} \frac{{C_{ij}}^2}{k_{ij}} = 0$$

This implies:

$$\lambda = \frac{A_j}{\sum_i^n \frac{C_{ij}^2}{k_{ij}}} \ \text{And} \ e_{ij}^* = \left(\frac{C_{ij}}{k_{ij}} \right) \left(\frac{\sum_i^n \left(a_{ij} - A_{ij} \right)}{\sum_i^n \frac{C_{ij}^2}{k_{ij}}} \right)$$

References

1. Hall E, Ken J, Jeremy D (2005) Requirement engineering, vol 3. Springer, London
2. INCOSE (2012) Guide for writing requirements. The International Council of Systems Engineering
3. Robertson S, Robertson J (2012) Measuring the requirement engineering process, getting the requirements right, 3d edn. Wesley
4. Hausmann JH, Heckel R, Taentzer G (2002) Detection of conflicting functional requirements in a use case-driven approach. 24th IEEE international conference on software engineering. ICSE
5. Fazelpour M Developing unit cell design guidelines for meso-scale periodic cellular materials. PhD Dissertation. Clemson University. December 2016
6. Eben KG, Lindemann U (2010) Structural analysis of requirements, Interpretation of structural criterions. International Dependency and Structure modeling Conference, UK
7. Fazelpour M, Summers J D (2013) A comparison of design approaches to meso-structure development. ASME 2013 international design engineering technical conferences and computers and information in engineering conference. American Society of Mechanical Engineers, Paper No. DETC2013-12295, p V03AT03A052
8. Flanigan D, Brouse P (2012) System of Systems requirements capacity allocation. SCER 112–117
9. Shankar P, Fazelpour M, Summers JD (2015) Comparative study of optimization techniques in sizing mesostructures for use in NonPneumatic tires. J Comput Inf Sci Eng 15(4):041009
10. Juristo N, Moreno AM, Silva A (2002) Is the European industry moving toward solving requirements engineering problems? IEEE Softw XX:70–77
11. Firesmith D (2007) Common requirements problems, their negative consequences, and the industry best practices to help solve them. J Object Technol 6(1):17–33
12. Fazelpour M, Summers JD (2014) Evolution of meso-structures for non-pneumatic tire development: a case study. ASME 2014 international design engineering technical conferences and computers and information in engineering conference. American Society of Mechanical Engineers, Paper No. DETC2014-34184, V02BT03A002
13. Yoder M, Satterfield Z, Fazelpour M, Summers JD, Fadel G (2015) Numerical Methods for the design of meso-structures: a comparative review. ASME 2015 international design engineering technical conferences and computers and information in engineering conference, Paper No. DETC2015-46289, p V02BT03A003

14. Emam KEL, Madhavji NH (1995) A Field studies of requirements engineering practices in information systems development. IEEE Int. Symp. On requirements engineering 2:68–80
15. Hall T, Beecham S, Rainer A (2002) Requirements problems in twelve software companies: an empirical analysis. Conference on empirical assessment in software engineering, 1–17
16. Hofmann HF, Lehner F (2011) Requirements engineering as a success factor in software projects. IEEE Softw 18:58–66
17. Fazelpour M, Shankar P, Summers JD (2016) Developing design guidelines for meso-scaled periodic cellular material structures under shear loading. ASME 2016 international design engineering technical conferences and computers and information in engineering conference, Paper No. DETC2016-59082
18. DUJDQ J, Compose-Nanez E, Fomin P, Wasek J (2014) Predicting system performance through requirements quality attributes model. Procedia Computer Science 28:347–353
19. Ke-shi Z, Wei-ji L, Hong-yan W (2006) A new method for optimum allocation of design requirements in aircraft conceptual design. Chinese journal of aeronaut 19:204–211
20. Gonzalo G, Fuentes J, Liorens J, Hurtado O, Moreno V (2011) A framework to measure and improve the quality of textual requirements. Requir Eng 18:25–24
21. Matsui K (2016) Asymmetric product distribution between symmetric manufacturers using dual-channel supply chain. Eur J Oper Res 248:646–657
22. Ping J, Xin M, Gang L (2015) Developing green purchasing relationships for the manufacturing industry: an evolutionary game theory perspective. Int J Prod Econ 166:155–162
23. Ma P, Wang H, Shang J (2013) Contract design for two-stage supply chain coordination: integrating manufacturer-quality and retailer-marketing efforts. Int J Prod Econ 146:745–755
24. Ghosh D, Shah J (2015) Supply chain analysis under green sensitive consumer demand and cost sharing contract. Int J Prod Econ 164:319–329
25. Örsdemir A, Kemahlıoğlu-Ziya E, Parlaktürk K (2014) Competitive quality choice and remanufacturing. Production and Operations Management Journal 1:48–64
26. Parlaktürk A (2012) The value of product variety when selling to strategic consumers. Manufacturing & Service Operations Management 3:371–385
27. Hazelrigg GA (1998) A framework for decision-based engineering design. J Mech Des 4:653–658
28. Lin YT, Chen YJ (2014) Competitive outsourcing: choosing between value-added services and key component supplying capability. To appear in International Journal of Production Research
29. Lee B, Paredis JJC (2014) A conceptual framework for value driven design and systems engineering. 24th CIRP design conference, 10–17
30. Collopy P, Hollingsworth P (2009)Value-driven design. 9th AIAA Aviation Technology, Integration, and Operations Conference (ATIO). 21–23
31. Collopy P, Otero J (2005) Value-driven design. Value -driven design workshop, Hartford

Chapter 64
Structural Rules for Sound Business Process Implemented by UML Activity Diagram

Mohanad A. Ajina, Bahram Yousefi, and Abbas K. Zaidi

Abstract A Business Process (BP) is defined to be a set of activities and tasks that is undertaken in some specified partial order to achieve an organizational goal. The validation of a BP is crucial for this goal to be achieved. The Unified Modeling Language (UML) Activity Diagram (ACT) is one of several ways of modeling a BP. Petri Nets (PNs) offer a more formal BP modeling language that also supports validation. A special class of PNs, called Workflow-Nets (WFNs), is specifically used to define a *sound* BP that is *deadlock-free* and *bounded*. The application of this definition requires a BP in the UML ACT form to be converted to a WFN. There are several proposed transformation algorithms from an ACT to an equivalent WFN representation in the literature. However, transforming a BP to an equivalent WFN requires knowledge of both the ACT and the WFN formalisms, and the transformation is also time consuming. This paper proposes a set of three structural rules for an ACT model of a BP to ensure the soundness property. The rules also help identify the errors if soundness of a BP cannot be established. We show that these structural rules are necessary for a sound BP modeled by the UML ACT. An illustrative example is also presented.

Keywords Activity Diagram • Business Process • Workflow-Net • Petri Net • Sound • Soundness • Unified Modeling Language

M.A. Ajina (✉)
Electrical and Computer Engineering Department, George Mason University, Fairfax, VA 22030, USA
e-mail: majina@gmu.edu

B. Yousefi • A.K. Zaidi
Systems Engineering and Operations Research, George Mason University, Fairfax, VA 22030, USA
e-mail: byousefi@gmu.edu; szaidi2@gmu.edu

A.M. Madni et al. (eds.), *Disciplinary Convergence in Systems Engineering Research*, https://doi.org/10.1007/978-3-319-62217-0_64

64.1 Introduction

Business Processes (BPs) are used in various engineering fields [1], and the validation of a BP is crucial for the goal to be achieved. The Unified Modeling Language (UML) Activity Diagram (ACT) is one of several ways of modeling the BPs. The UML is used to model the static and dynamic aspects of complex systems [2]. It provides industry standard mechanisms for visualizing, specifying, analyzing, designing, constructing, and documenting software systems [3], as well as for modeling BPs and similar workflows [4]. Unfortunately, the execution semantics for UML is not sufficiently precise to be unambiguous, which is broadly acknowledged [5].

Therefore, practitioners tend to transform the BPs to more formal modeling languages for further analyses, such as Petri Net (PN). There are many PN's classes to model the BPs [6]. Wikarski, Han, and Löwe in [7] introduced a Higher Order Object Nets class of PN to model the BPs. Aalst in [8] introduced a special class of PN to model the BPs that is called Workflow-Net (WFN). The advantage of WFN is that it provides a formal definition of a *sound* BP. Also, there are several simulation softwares that can verify the soundness properties, but this is not the case for an ACT. On the other hand, the transforming of a BP to an equivalent WFN is time consuming and requires knowledge of both the ACT and the WFN formalisms.

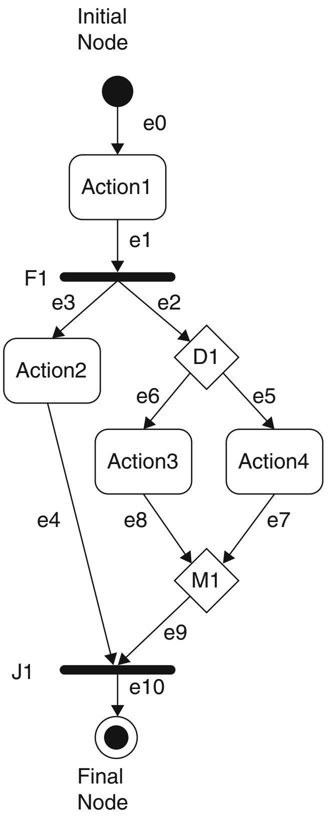
In this paper, we introduce structural rules to model a *sound* BP implemented by the UML ACT. We hypothesize that these rules can verify the soundness of a BP and can also help identify the errors if soundness cannot be established. To test our hypothesis, we developed Path Conservation Techniques (PCTs) that utilize structural reductions of an ACT to a simpler structure with an equivalent *sound* WFN. We showed that these structural rules are necessary for a *sound* BP implemented by an ACT. Also, the approach can be utilized for ensuring the correctness of the BP modeled with the help of the ACT without the need of the WFN.

The paper is organized as follows: Section 64.2 will introduce modeling formalism of a BP and an ACT. Sections 64.3 and 64.4 will explain the Structural Rules and the PCTs, respectively. Section 64.5 will provide an example of a BP as an illustration of the proposed approach. Lastly, Section 64.6 will be the conclusion.

64.2 Business Process and Activity Diagram

In this section, we discuss the modeling formalism of a BP and a UML ACT. The BP is defined in [9] as "an ordered set of actions or activities, linked by precedence relationships, and triggered by an event that terminates in some observable end result, which achieves some business goal. It represents the flow and control of things happening in the enterprise." Figure 64.1 represents an example of a BP implemented by the ACT. The BP in Fig. 64.1 is a graph containing vertices

Fig. 64.1 A BP modeled using most common node in an ACT, where "F1" node is a Fork Node, "J1" is a Join Node, "Action1", . . ., "Action4" nodes are Action Nodes, "D1" node is a Decision Node, and "M1" node is a Merge Node



(nodes), edges, and paths. In this paper, the BPs implemented by an ACT are directed and simple graphs.

Definition 1 (Graph) [10]: A graph G is a set of vertices (nodes) V(G) connected by edges E(G). Thus, G = (V(G), E(G)).

Definition 2 (Vertices) [10]: A vertices $v\in$ V(G) is a terminal point or an intersection point of a graph.

Definition 3 (Edge) [10]:An edge $e\in$ E(G) is a link between two nodes. The edge (i, j) is of initial extremity i and of terminal extremity j.

Definition 4 (Path) [11]: A Path is a simple graph whose vertices can be ordered so that two vertices are adjacent if and only if they are consecutive in the ordering. A Path that begins at vertex v_1 and ends at vertex v_2 is called a (v_1, v_2) path.

Definition5 (Directed Graph) [10]: A graph G is a directed graph if and only if all e $\in$ E(G) specify the direction of the flow.

Definition6 (Simple Graph) [11]: A graph G is a simple graph if the graph contains neither loops nor multiple edges going in or out of an *Action Node*.

Definition 7 (Subgraph) [11]: A subgraph G1 of a graph G0 is a graph such that V (G1) $\subseteq$ V (G0) and E(G1) $\subseteq$ E(G0), satisfying the property that for every $e\in$ E

(G1), where e has endpoints $v_1, v_2 \in$ V (G0) in the graph G0, then $v_1, v_2 \in$ V (G1) and e has endpoints v_1, v_2 in G1. Note the edge relation in G1 is the same as in G0.

The BP model in Fig. 64.1 is a simple and direct graph that can be represented as follows:

- BP = (V (BP), E (BP))
- V (BP) = (Initial Node, Action1, Action2, Action3, Action4, D1, M1, F1, J1, Final Node)
- E (BP) = e_0, e_1, e_2, e_3, e_4, e_5, e_6, e_7, e_8, e_9, e_{10}
- Path1 (BP) = (Initial Node, Action1, F1, D1, Action3, J1, Final Node), this is just one example of a path.

The ACT is a step-by-step graphical representation that shows the flow of token from one step to another [4, 12]. The token begins travelling from an Activity Start Node (ASN) until it reaches an Activity Final Node (AFN). When a token reaches the AFN, the process stops, and all the remaining tokens in the process are discarded. The ACT graphical notations are defined in more detail as follows [12]:

Definition 8 (Activity Start Node): An ASN is the starting point for the process.
Definition 9 (Activity Final Node): An AFN is the end point for the process.
Definition 10 (Action Node): An Action Node (AN) is a step in the overall activity.
Definition 11 (Decision Node): A Decision Node (DN) is where the exit transition from a state or activity may branch in alternative flows.
Definition 12 (Merge Node): A Merge Node (MN) brings together alternate flows into a single output flow (Note that it does not synchronize multiple concurrent flows).
Definition 13 (Fork Node): A Fork Node (FN) splits the current flow through an activity into multiple concurrent flows.
Definition 14 (Join Node): A Join Node (JN) synchronizes multiple flows of an activity back to a single flow of execution.
Definition 15 (Control Flow): A Control Flow (CF) explicitly models control passing from one action to the next.

Each graphic construct of an ACT has its own operation semantics. The following are some graphic constructs of the ACT [4]:

- A black circle represents an Activity Start Node.
- An encircled black circle represents an Activity Final Node.
- A rounded rectangle represents an Action Node.
- A bar represents a Fork Node or a Join Node.
- A diamond represents a Decision Node or a Merge Node.
- A solid arrow head represents a Control Flow.

64.3 Structural Rules for a Sound Business Process

In this section, we introduce our structural rules to model a *sound* BP implemented by an ACT. The structural rules map the WFN properties for a *sound* BP to a set of three structural rules for the ACT. A BP implemented by a WFN is *sound* if and only if a *strongly connected* WFN is *bounded* and *Deadlock-free*. A deadlock-free WFN is defined as a *Live* WFN [8]. We begin by briefly introducing the PN and the WFN.

Definition 16 (Petri Nets) [8]: Petri Net is a triple tuple, PN = (P, T, F) where
$P \neq \emptyset$ is a finite set of place nodes
$T \neq \emptyset$ is a finite set of transition nodes ($P \cap T = \emptyset$)
$F \subseteq (P \times T) \cup (T \times P)$ is a set of arcs (flow relation)
Definition 17 (Workflow-Net) [8]: A PN = (P, T, F) is a WFN if and only if PN has two special places: *i* and *o*. Place i is a source place: $\bullet i = \emptyset$. Place *o* is a sink place: $o\bullet = \emptyset$.

Definition 18 (Strongly Connected WFN) [8]: A WFN is strongly connected if and only if there is a transition *t** to WFN, which connects place *o* with *i*.
Definition 19 (Bounded) [8]: A WFN is bounded if and only if, for every reachable state and every place *p*, the number of tokens in *p* is bounded.
Definition 20 (Live) [8]: A WFN is live if and only if, for every reachable state *M0* and every transition *t*, there is a state *M1* reachable from *M0*, which enables *t*.

To better understand the structural errors of an ACT, we shall explain some of them. One structural error in an ACT is that the ACT has two or more *concurrent paths* ending at an AFN, as shown in Fig. 64.2a. The ACT will have at least one incomplete task in the process, an incomplete error. When a token from any *concurrent path* reaches the AFN, the other tokens in the process are discarded whether the process is completed or not. In the equivalent strongly connected WFN, this error will result in unbounded WFN. Another error in an ACT is that the ACT

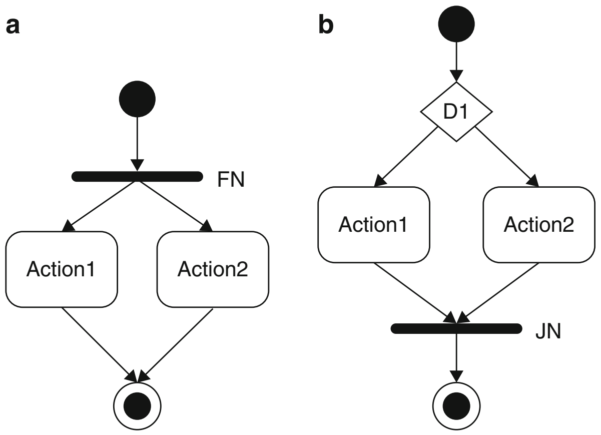


Fig. 64.2 (**a**) An incomplete process error in the ACT, and (**b**) A hanging token error in the ACT

has two or more *alternative paths* ending at a JN, as shown in Fig. 64.2b. This ACT will have a hanging token in the process, because connecting *alternative paths* to a JN will not provide the required tokens for the JN to execute. In the equivalent strongly connected WFN, this error will result in a deadlock, not live WFN. These errors and many others will be resolved by applying the proposed structural rules. To simplify our approach, we introduce the following definitions.

Definition 21 (Initial Node of Subgraph (INS)): A node v_1 is an INS G1 if the node is a DN or a FN.

Definition 22 (Subgraph with Concurrent Paths (SCP)): A SCP, G1, of a graph G0 is a graph such that the INS G1 is an FN.

Definition 23 (Subgraph with Alternative Paths (SAP)): A SAP, G1, of a graph G0 is a graph such that the INS G1 is a DN.

Definition 24 (Concurrent Paths): The concurrent paths are the paths starting from a FN.

Definition 25 (Alternative Paths): The alternative paths are the paths starting from a DN.

Figure 64.3 shows initial examples of SCP and SAP. In an advanced BP model, the BP is larger and may contains many of complex SCPs and SAPs. In addition, the validation happens to be more complex if the SCP is a subgraph of SAP or vice versa.

Our structural rules that are mapped from the soundness properties of the WFN are defined as follows:

I. All paths in a SCP of a SAP must be connected to a JN before merging or synchronizing with other paths.
II. All paths in a SAP of a SCP must be connected to a MN before synchronizing or merging with other paths.
III. All paths in a SCP of graph G0 must be connected to a JN before connecting to an AFN.

Note: To simplify our graphs, we will assume the arrow heads represent a sequence of ANs or a single CF.

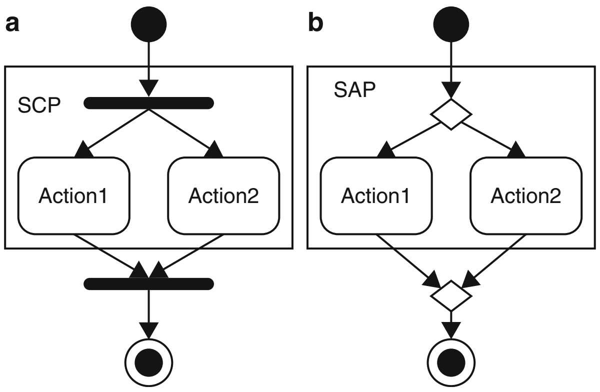


Fig. 64.3 (**a**) An ACT with SCP, and (**b**) An ACT with SAP

The structural rule I implies that all concurrent paths of a SCP of a SAP must be synchronized together as illustrated in Fig. 64.4. The SCP establishes two or more concurrent paths. If these paths are merged, not synchronized, then only one token will reach the AFN, and the other tokens will be terminated. In addition, if these paths are synchronized with paths outside the SPC, then it is not guaranteed that the JN will execute. This rule ensures all JNs are executable, ensures completing the process for all possible paths. Accordingly, applying this rule will result in *Closed Subgraph with Concurrent Paths* (CSCP) as shown in Fig. 64.5.

Definition 26 (Closed Subgraph with Concurrent Paths (CSCP)): A SCP, G1, of a graph G0 is a CSCP if and only if all paths in the SCP are mapped to a JN.

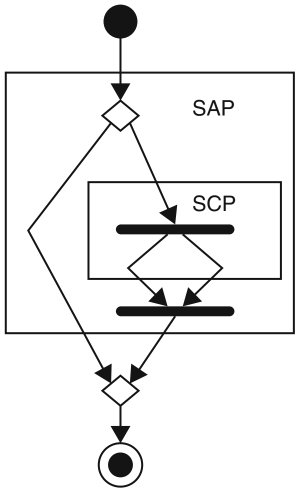


Fig. 64.4 An implementation of structural rules I

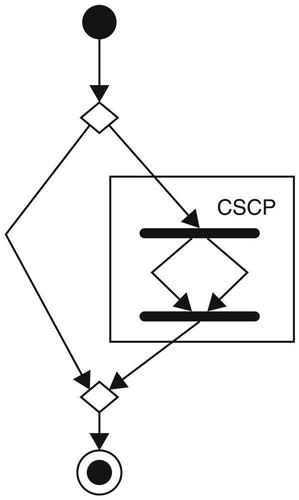


Fig. 64.5 A closed subgraph with alternative paths

The structural rule II implies that all the alternative paths of a SAP of a SCP must be merged together as illustrated in Fig. 64.6. The SAP establishes two or more alternative paths, but only one path can have a token. If these paths are synchronized, then the JN will not execute because of missing tokens. In addition, if these paths are merged with paths outside the SAC, then it is not guaranteed the merged paths produce one token only to the AFN. If the alternative paths have more than one token, this implies incomplete process error. This rule ensures that each SAP produce one token only, and ensures completing the process for all possible paths. Accordingly, applying this rule will result in a *Closed Subgraph with Alternative Paths* (CSAP) as shown in Fig. 64.7.

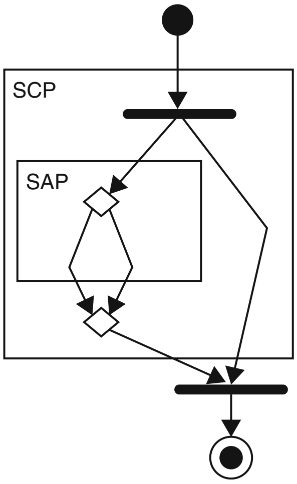


Fig. 64.6 An implementation of structural rules II

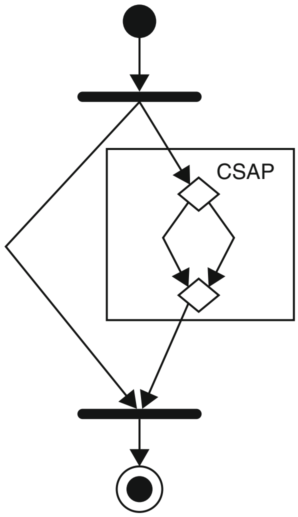


Fig. 64.7 A closed subgraph with concurrent paths

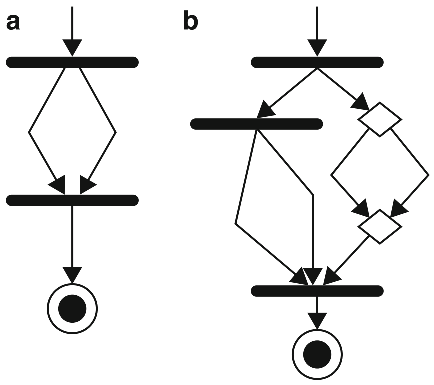


Fig. 64.8 (**a**) An implementation of structural rule III on a simple ACT, and (**b**) An implementation of structural rule III on a more complex ACT

Definition 27 (Closed Subgraph with Alternative Paths (CSAP)): A SAP, G1, of a graph G0 is a CSAP subgraph with alternative paths if and only if all paths in the SAP are mapped to a MN or an AFN.

The structural rule III synchronizes all paths of a SCP to ensure completing the process. Figure 64.8a, b shows examples of a simple and a more complex ACT, respectively, that illustrate this rule. On the other side, alternative paths in a SAP may or may not be merged before connecting to an AFN because they only produce one token. More explanation is provided in Sect. 64.4.

64.4 Path Conservation Techniques

In this section, we will introduce Path Conservation Techniques (PCTs) to validate a BP implemented by an ACT. The PCTs utilize structural reductions of the ACT to a simpler structure with an equivalent sound WFN. The PCTs are as follows:

I. Replace every sequence of ANs by a CF.
II. Replace every CSCP that does not contain any SAP by a CF.
III. Replace every CSAP that does not contain any SCP by a CF.
IV. Replace all alternative paths that end at an AFN by a CF.

The PCT I is simple to show. Since a sequence of ANs does not produce new concurrent or alternative paths, the flow of a token from a start node to an end node can be shown by one CF. Figure 64.9a shows an example of a sequence of ANs, and Fig. 64.9b shows the reduced graph of Fig. 64.9a.

Note: in this section, Node1 could be an ASN, a DN, or a FN, and Node2 could be an AFN, a MN, or a JN.

The PCT II and PCT III could be applied to a simple or a complex structure of a CSCP and a CSAP, respectively. These techniques must be repeated until all

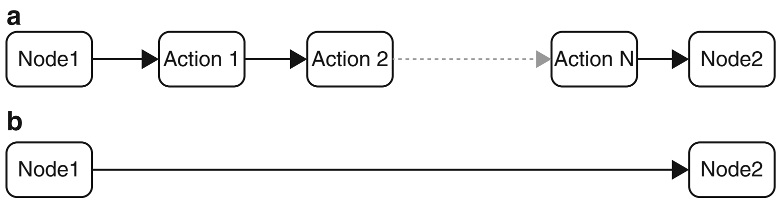


Fig. 64.9 (**a**) A sequence of Action Nodes, and (**b**) A reduced graph of (**a**)

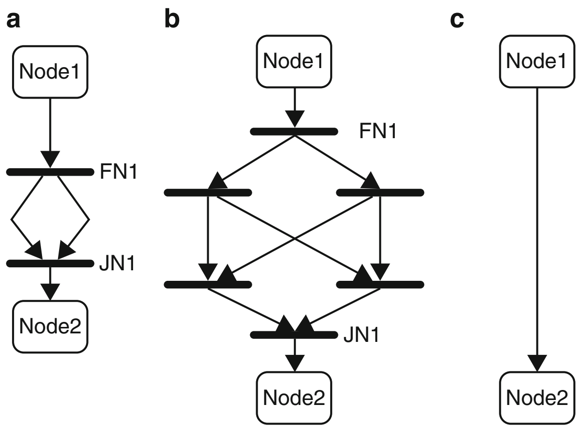


Fig. 64.10 (**a**) A simple structure of a CSCP, (**b**) A complex structure of a CSCP, and (**c**) The reduced structure of the figure (**a**) and (**b**)

CSCP and CSAP are replaced with CFs in the ACT. Assuming the PCT I is applied, Fig. 64.10a, b shows the simple and complex structures of a CSCP. Figure 64.11a, b shows the simple and complex structures of a CSAP. These techniques simplify the paths that start at Node1 and end at Node2. For any chosen path in the ACTs, the path will always have the same starting node, Node1, and the same ending node, Node2. Thus, the direction of a token flowing in the CSCP or the CSAP, can be replaced by one CF. Figure 64.10c shows the reduced structure of Fig. 64.10a, b. Figure 64.11c shows the reduced structure of Fig. 64.11a, b.

The PCT IV is implemented to include the last case where there are multiple alternative paths ending at an AFN. This technique can only be applied last. Figure 64.12a, b shows a simple and a complex example with multiple alternative paths ending at the AFN, respectively. Since all paths are alternative, there is only one token in the ACT that could reach the AFN. Thus, the direction of the flowing token from the ASN to the AFN can be replaced by one CF. The reduced structure of Fig. 64.12a, b is shown in Fig. 64.13.

The reduced ACT, after applying the four PCTs, is shown in Fig. 64.13. The reduced ACT contains only an ASN and an AFN mapped by one CF. This ACT will always have an equivalent sound WFN. If a BP implemented by an ACT can be reduced by the PCTs to the ACT in Fig. 64.13, then the BP is sound. We shall illustrate the structural rule and the PCTs on a BP example in the following section.

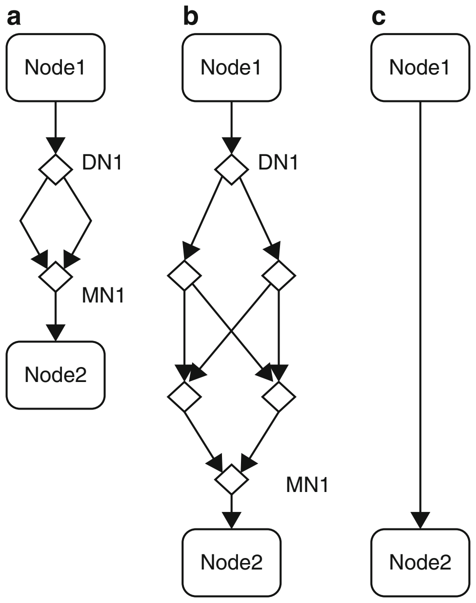


Fig. 64.11 (**a**) A simple structure of a CSAP, (**b**) A complex structure of a CSAP, and (**c**) The reduced structure of the figure (**a**) and (**b**)

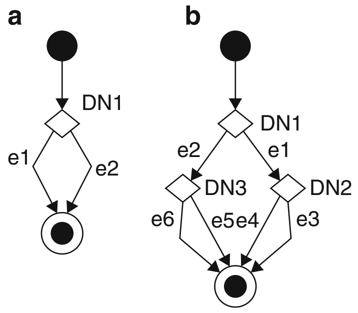


Fig. 64.12 (**a**) An example of an ACT with multiple paths ending at the AFN, and (**b**) An example of an ACT with multiple paths and paths ending at the AFN

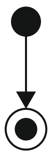


Fig. 64.13 The reduced structure of the ACT after applying the PCTs

64.5 Example

In this section, we provide an example of a BP implemented by the ACT to illustrate the proposed approach. The example is shown in Fig. 64.14, and it is adopted from the process of obtaining the United States Naturalization Certificate – this BP is available online [13]. The BP has two SCPs and two SAPs. This BP

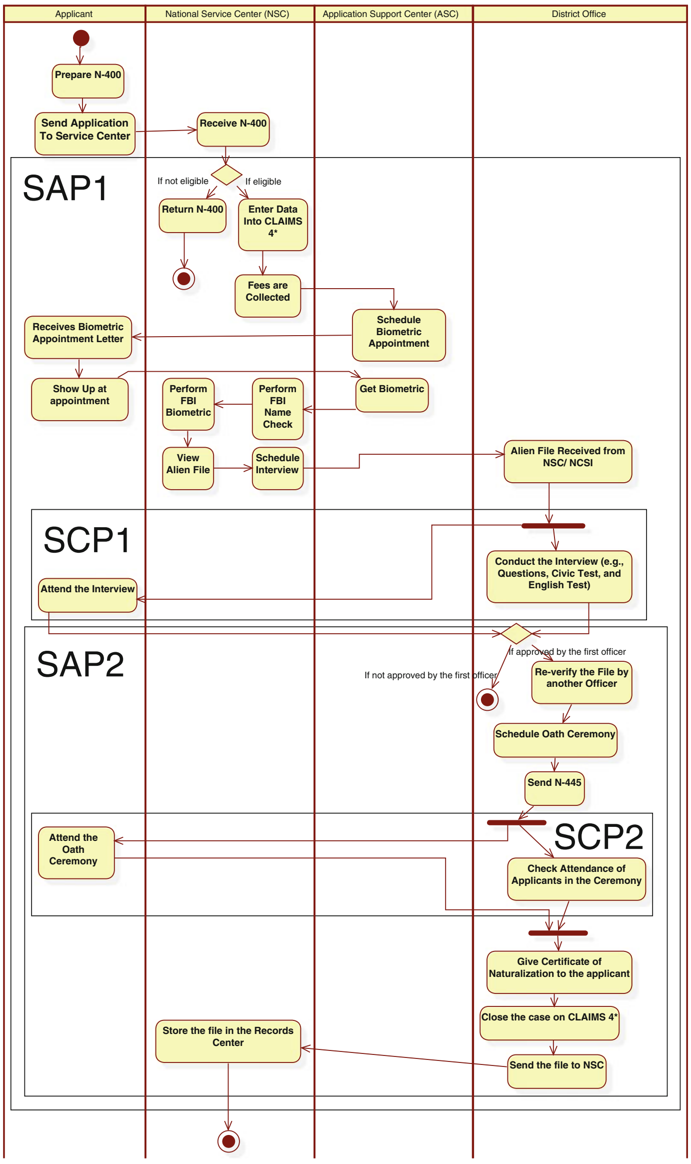

Applicant
National Service Center (NSC)
Application Support Center (ASC)
District Office
Prepare N-400
Send Application To Service Center
Receive N-400
SAP1
If not eligible
If eligible
Return N-400
Enter Data Into CLAIMS 4*
Fees are Collected
Schedule Biometric Appointment
Receives Biometric Appointment Letter
Show Up at appointment
Perform FBI Biometric
Perform FBI Name Check
Get Biometric
View Alien File
Schedule Interview
Alien File Received from NSC/ NCSI
SCP1
Conduct the Interview (e.g., Questions, Civic Test, and English Test)
Attend the Interview
SAP2
If approved by the first officer
If not approved by the first officer
Re-verify the File by another Officer
Schedule Oath Ceremony
Send N-445
SCP2
Attend the Oath Ceremony
Check Attendance of Applicants in the Ceremony
Give Certificate of Naturalization to the applicant
Close the case on CLAIMS 4*
Store the file in the Records Center
Send the file to NSC

Fig. 64.14 An illustrative example of the proposed approach

invalidates our structural rules. We can observe that if an application is not approved by the first officer, then some tokens will be discarded. When "Alien File Received from NSC/NCSI" is executed, it produces a token for "Attend Interview" and a token for "Conduct Interview." One of these produced tokens will be discarded if one token flows to the MN sooner than the other one; thus, there will be an incomplete error. This BP also has other errors depending on the excitation sequence of the BP. All errors will be resolved by applying our structural rules as shown in Fig. 64.15. We verified our structural rules by identifying all CSAP and CSCP. To validate the soundness of the BP, we shall apply the PCTs:

- Step1: Verify the structural rules as shown in Fig. 64.15.
- Step 2: Replace all sequences of ANs by CFs. The reduced ACT is shown in Fig. 64.16. Readers may notice that the subgraphs remain the same.
- Step 3: Replace both CSCP1 and CSCP2 by two CFs. The reduced ACT is shown in Fig. 64.17.
- Step 4. Replace the CSAP2 by a CF. The reduced ACT is shown in Fig. 64.18.
- Step 5: Replace CSAP1 by a CF. The reduced ACT is shown in Fig. 64.19. Therefore, this BP model is sound because it is reduced to the smallest structure that has the equivalent sound WFN.

Since the BP is reduced to the smallest ACT by the PCTs, this BP is sound. For further verification of the soundness properties of the BP in Fig. 64.15, we transform the ACT to a WFN using the transformation provided in [14]. Figure 64.20 shows the strongly connected WFN of the BP. Figure 64.21 shows the soundness properties result of the BP. As a result, the BP model is deadlock-free and bounded, which implies that the BP is sound.

64.6 Conclusion

In this paper, our goal is to model a sound BP implemented by an ACT**.** Subsequently, we introduced a set of three structural rules for the UML ACT, which is mapped from the soundness properties of the Workflow-Nets (WFNs). We enhanced our approach by developing PCTs that utilize structural reductions of an ACT to a simpler structure with an equivalent sound WFN. We showed that these structural rules are necessary for a sound BP implemented by an ACT. Also, we provided a real-world example to illustrate our approach and showed that applying the structural rules resolve the error or errors that a BP may have. Our approach can be utilized for ensuring correctness of BPs modeled by the UML ACTs without the need of WFNs.

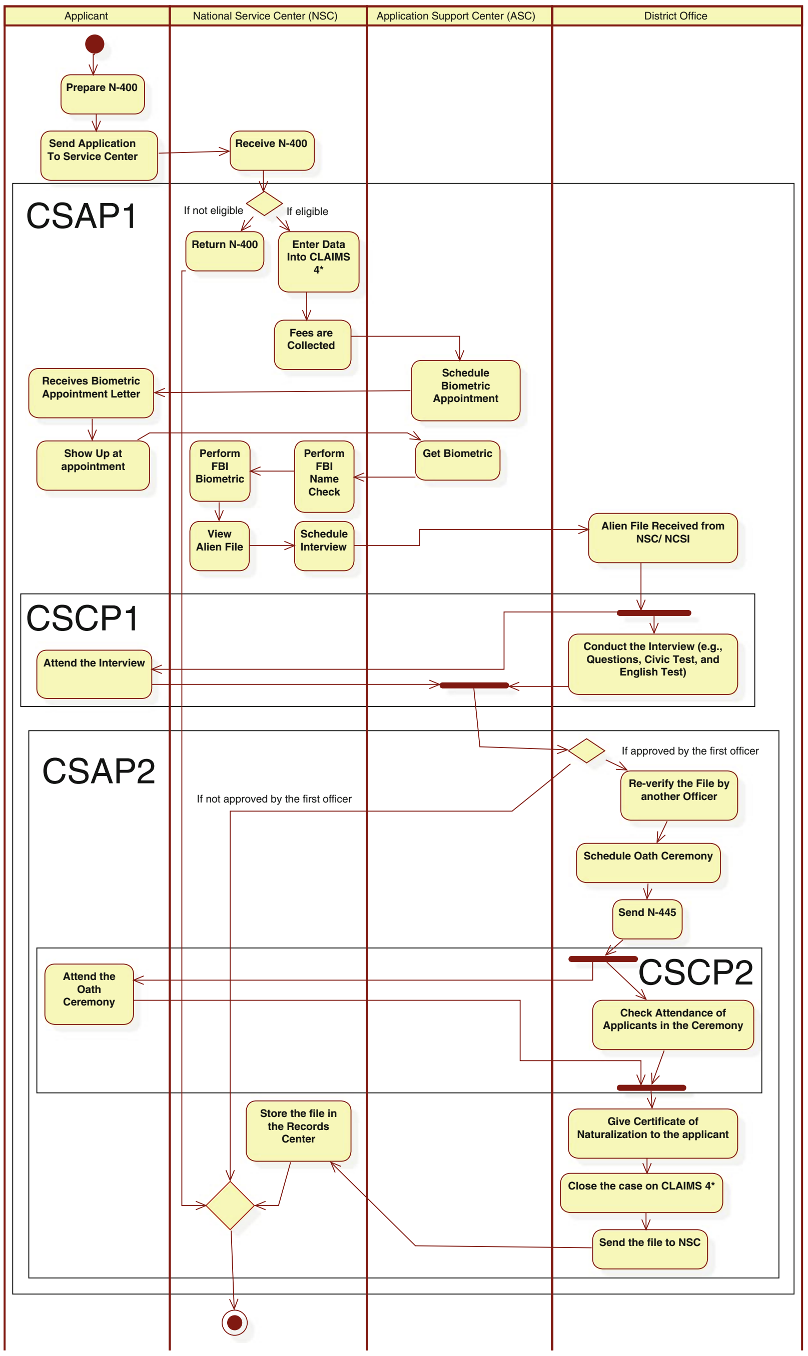

Applicant
National Service Center (NSC)
Application Support Center (ASC)
District Office
Prepare N-400
Send Application To Service Center
Receive N-400
CSAP1
If not eligible
If eligible
Return N-400
Enter Data Into CLAIMS 4*
Fees are Collected
Schedule Biometric Appointment
Receives Biometric Appointment Letter
Show Up at appointment
Perform FBI Biometric
Perform FBI Name Check
Get Biometric
View Alien File
Schedule Interview
Alien File Received from NSC/ NCSI
CSCP1
Attend the Interview
Conduct the Interview (e.g., Questions, Civic Test, and English Test)
CSAP2
If approved by the first officer
If not approved by the first officer
Re-verify the File by another Officer
Schedule Oath Ceremony
Send N-445
CSCP2
Attend the Oath Ceremony
Check Attendance of Applicants in the Ceremony
Give Certificate of Naturalization to the applicant
Store the file in the Records Center
Close the case on CLAIMS 4*
Send the file to NSC

Fig. 64.15 The resolved BP in Fig. 64.14

Fig. 64.16 The BP after applying PCT I in step 2

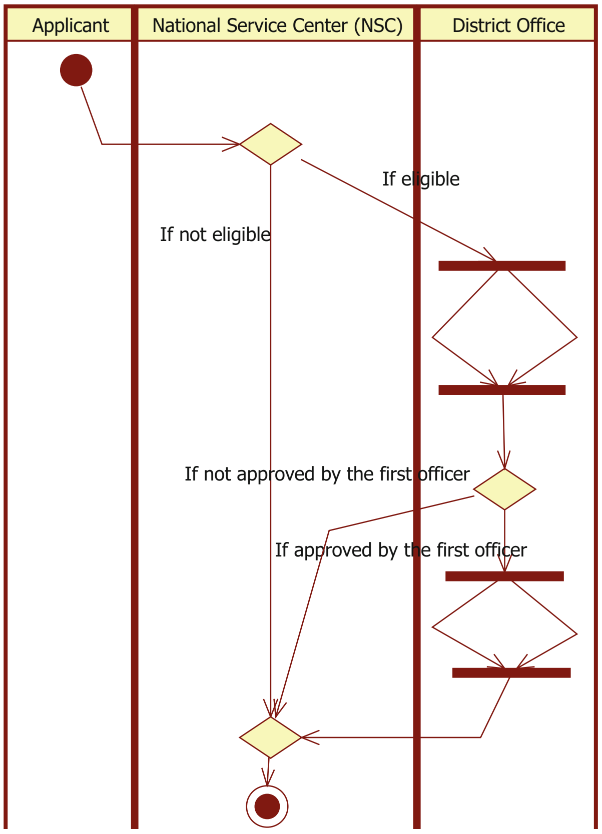
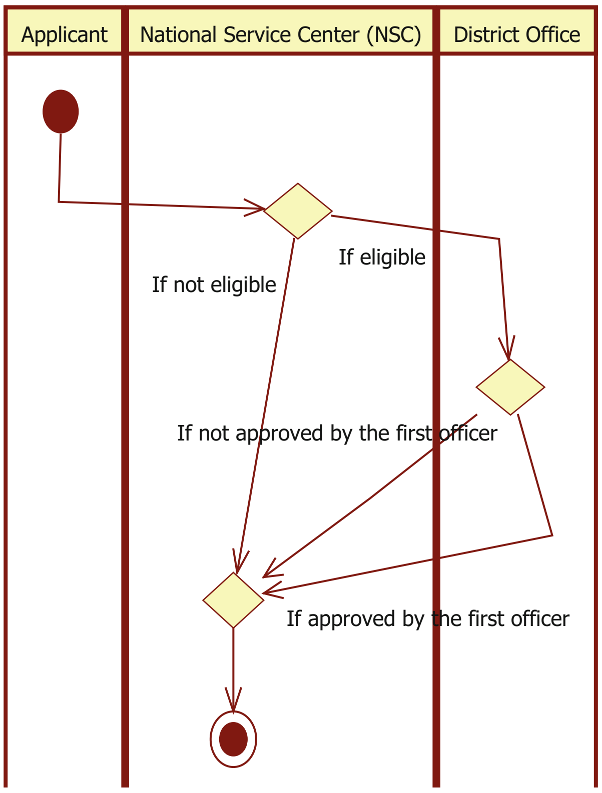

Applicant
National Service Center (NSC)
District Office
If eligible
If not eligible
If not approved by the first officer
If approved by the first officer

Fig. 64.17 The BP after applying PCT II in step 3


Applicant
National Service Center (NSC)
District Office
If eligible
If not eligible
If not approved by the first officer
If approved by the first officer

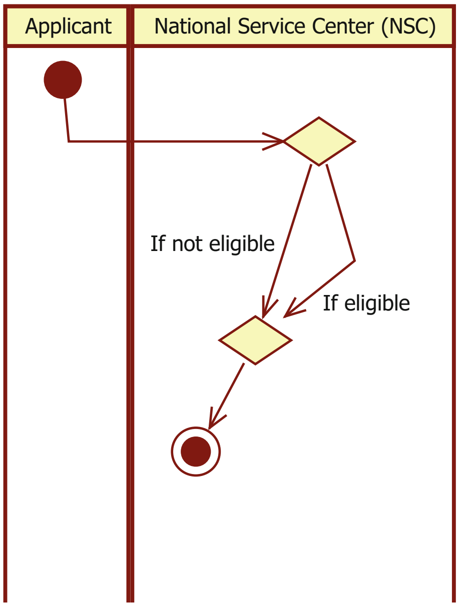


Fig. 64.18 The BP after applying PCT III in step 4

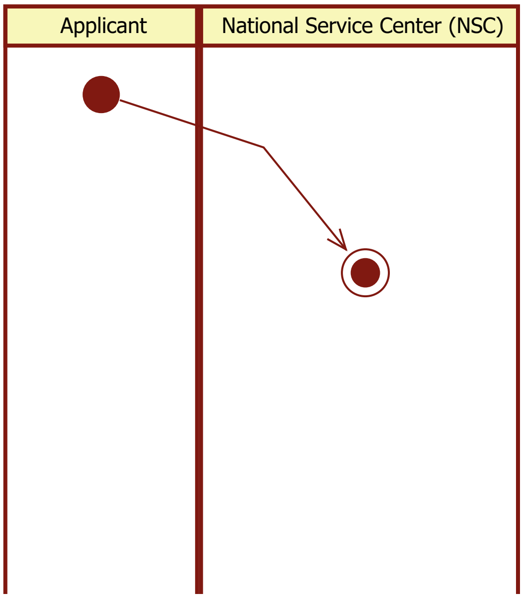


Fig. 64.19 BP reduced to the smallest structure that is equivalent to a sound WFN

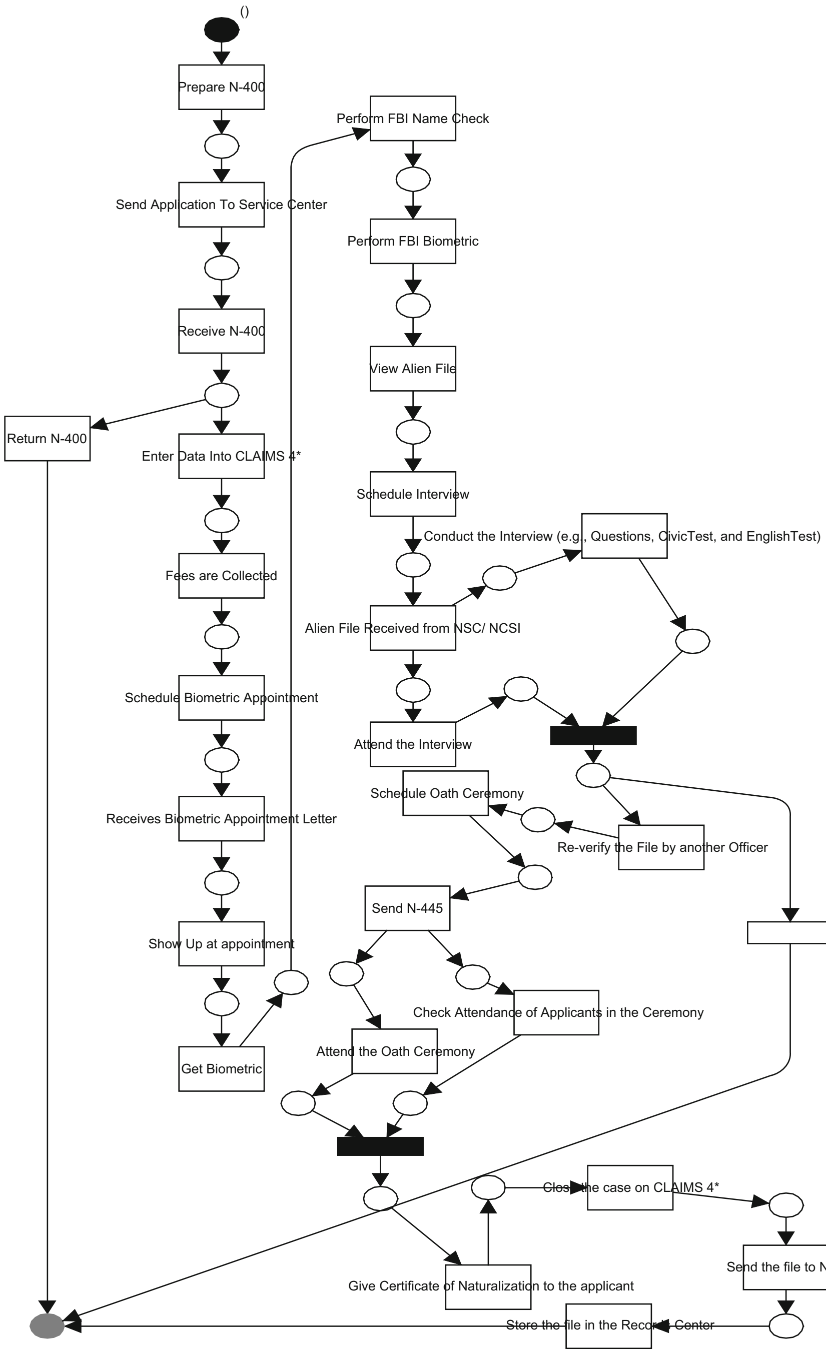
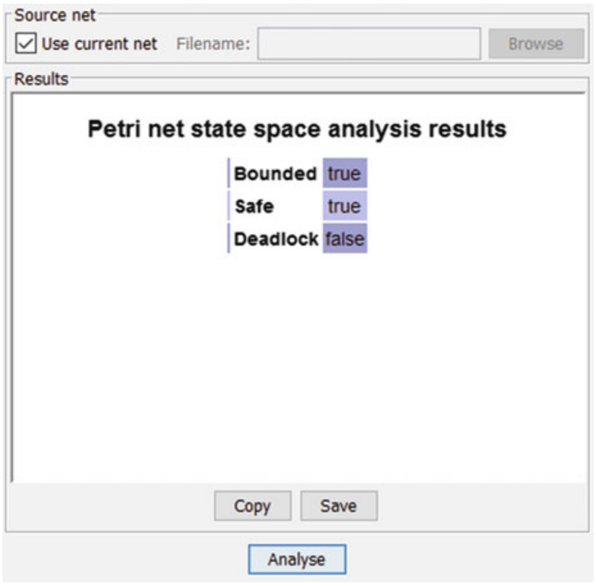

()
Prepare N-400
Send Application To Service Center
Receive N-400
Return N-400
Enter Data Into CLAIMS 4*
Fees are Collected
Schedule Biometric Appointment
Receives Biometric Appointment Letter
Show Up at appointment
Get Biometric
Perform FBI Name Check
Perform FBI Biometric
View Alien File
Schedule Interview
Conduct the Interview (e.g., Questions, CivicTest, and EnglishTest)
Alien File Received from NSC/ NCSI
Attend the Interview
Schedule Oath Ceremony
Re-verify the File by another Officer
Send N-445
Check Attendance of Applicants in the Ceremony
Attend the Oath Ceremony
Close the case on CLAIMS 4*
Send the file to NSC
Give Certificate of Naturalization to the applicant
Store the file in the Records Center

Fig. 64.20 The strongly connected WFN of the BP implemented in Fig. 64.15



Petri net state space analysis results

Bounded	true
Safe	true
Deadlock	false

Fig. 64.21 The soundness properties result

References

1. Weske M (2012) Business process management: concepts, languages, architectures. Springer, Berlin
2. Yang D, Wu H, Tong L (2009) A UML-based approach for the development of shop floor control systems. Int J Prod Res 47(6):1601–1633
3. Eriksson H (2004) UML 2 toolkit. Wiley Pub, Indianapolis
4. Unified Modeling Language Superstructure Specification V2.4.1, OMG, 2011
5. Mellor SJ, Balcer MJ (2002) Executable UML: a foundation for model-driven architecture. Addison Wesley Professional, Boston
6. van der Aalst W, Desel J, Oberweis A (2000) Business process management:models, techniques, and empirical studies, vol 1806. Springer, Berlin Heidelberg
7. Wikarski D, Han Y, Löwe M (1995) Higher-order object nets and their application to workflow modeling. Technische Universität Berlin, Forschungsberichte der FB Informatik, Berlin, pp 95–34
8. van der Aalst WMP (1997) Application and theory of petri nets. In: Azma P, Balbo G (eds) Verification of workflow nets, vol 1248 of Lecture Notes in Computer Science. Springer, Berlin, pp 407–426
9. Shtub A, Karni R (2009) ERP. Springer, New York, pp 31–57
10. Rodrigue J, Comtois C, Slack B (2006) The geography of transport systems. Routledge, London
11. Forrest J, Duan X, Zhao C, Xu L (2012) Systems science: methodological approaches. CRC Press, Boca Raton
12. UML activity diagrams are UML behavior diagrams which show flow of control or object flow with emphasis on the sequence and conditions of the flow., Uml-diagrams.org, 2016. [Online]. Available: http://www.uml-diagrams.org/activity-diagrams.html. Accessed 28 Apr 2016

13. Application for Naturalization – Form N-400 and Process | Lawfirms.com. [Online]. Available: http://www.lawfirms.com/resources/immigration-law/us-immigration-forms/naturalization-n400.htm. Accessed 22 Jan 2017
14. Staines TS (2008) Intuitive mapping of UML 2 activity diagrams into fundamental modeling concept petri net diagrams and colored petri nets, Engineering of Computer Based Systems,*2008*. ECBS 2008. 15th Annual IEEE international conference and workshop on the, Belfast, pp 191–200

Chapter 65
A Value-Driven Approach to Capture Unintended Consequences Impacting Mission Success

David Kis, Christopher Wenger, and Christina L. Bloebaum

Abstract Large-scale complex engineered systems are systems whose complexity and numerous inherent couplings can lead to unintended consequences or unanticipated behaviors during system operation. In many cases, these behaviors have negligible impact on system functionality; however, some unanticipated behaviors can contribute to a deficiency or even a failure of the system. Capturing these behaviors during the design and development phase (before testing and operation) can aid system managers in a reduction of time, costs, or even cancellation of projects due to a disastrous unforeseen interaction. Early identification also aids engineers to redesign components at an early phase of the development process, instead of relying on mitigation to address the issue after the fact. The goal of this paper is to use a coupling strength analysis from the field of multidisciplinary design optimization (MDO) to identify possible unanticipated behaviors or consequences due to interactions that were previously assumed of little importance. An evaluation of possible losses in value due to unforeseen behaviors will also be made using a value-based systems engineering (VBSE) approach.

Nomenclature

A_c	Chamber Cross-Sectional Area
DSM	Design Structure Matrix
$D_{x,y,z}$	Nodal Displacement
T/O	Thrust Oscillation
f_a	Acoustic Mode
f_s	Structural Mode
f_{vs}	Vortex Shedding Frequency
L_i	Length of Inhibitor
L_c	Chamber Length

D. Kis (✉) • C. Wenger • C.L. Bloebaum
Iowa State University, Ames, IA 50011, USA
e-mail: dkis@iastate.edu

A.M. Madni et al. (eds.), *Disciplinary Convergence in Systems Engineering Research*, https://doi.org/10.1007/978-3-319-62217-0_65

n_{nodes}	Total Number of Nodes
P_A	Ambient Pressure
P_c	Chamber Pressure
R_c	Chamber Radius
$S.T$	Strouhal Number
U_f	Velocity of Fluid
X	Design Variable
Y	Behavior Variable
i, j, n, m	Subsystem Index

65.1 Introduction and Motivation

In October 2009, NASA test launched the Ares-1X, a prototype of the Ares 1, and used as a launch vehicle for the Constellation Program. The intended mission for the Ares 1 was to launch the Orion, a crew module, into orbit to allow the crew to perform missions varying from work on the International Space Station to a manned mission to Mars [1]. During testing, large vibrations within the rocket engine were recorded late into the first stage flight, causing vibrations to propagate throughout the rocket and crew module. These vibrations were caused by a larger than average cyclical loading on the internal structure of the rocket engine known as thrust oscillation (T/O). The propagation of these loads posed a strong hazard to the function and health of the crew during a key period of operation.

In this paper, the T/O issue is investigated using a simplified model of a solid rocket motor (SRM) and then reproducing the effects that occurred on the Ares 1 using simulation. A Multidisciplinary Coupling Analysis (MCA) is proposed as a means by which critical subsystem feedbacks can be identified and evaluated to determine impact on system objectives. Further, once these interactions have been identified, a value model is then created to evaluate the importance of capturing this specific unintended consequence to mission success. To understand the importance of identifying critical couplings and its effect on value, the physics associated with the T/O event will be discussed.

65.2 Background

65.2.1 Thrust Oscillation

Thrust oscillation is a well-known phenomenon that occurs in solid rocket motors during flight and operation. This phenomenon is caused by combustion instabilities within the rocket engine that involves a coupling interaction between the acoustic modes, fluid flow, and structural mode responses [1]. During combustion, unstable shear layers are produced by sudden transitional changes in the flow field [2]. These

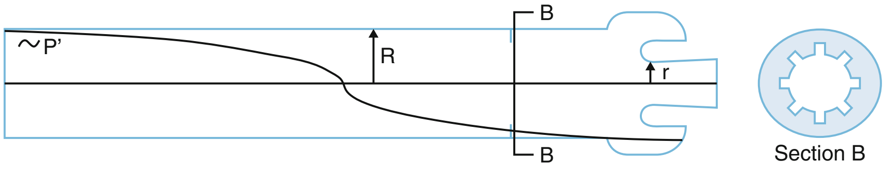


Fig. 65.1 Diagram of solid rocket motor with inhibitor cross section

sudden transitions cause turbulent flow, more specifically vortex shedding, to form downstream, which eventually interacts with the surface of the nozzle causing waves of pressure. These pressure waves oscillate throughout the engine chamber, reflecting and even propagating back upstream [3]. The waves of pressure, known as pressure oscillation, cause an increase of energy in the rocket cavity, which is translated to a force on the chamber walls. Typically, the excited energy caused by pressure oscillation is dampened out, causing no harm or instability to the rocket. In the case where the pressure oscillation is generated by vortex shedding, even greater excitation is noticed, which causes a greater structural response, known as T/O [2]. If T/O is not mitigated or dampened, the unstable vibrations caused in the engine chamber can propagate throughout the rocket, affecting both the crew on board and the structural supports. Ideally, the vehicle natural resonance should be a prescribed distance apart from pressure oscillation frequencies to avoid a major T/O event [1] (Fig. 65.1).

65.2.2 Solid Rocket Motor Physics

SRMs have multiple subsystems interacting with one another to complete the system's intended design objective. While there are many disciplines that interact in an SRM system analysis, this paper focuses on three key subsystems that have been previously identified to be the underlying drivers of T/O.

The three main subsystems investigated in this paper are acoustics, fluids dynamics, and structures. The means by which these are coupled is discussed later in this paper. In this section, we will provide an overview of the physics associated with each that drive the T/O event.

Acoustics is a subsystem that monitors the interactions of acoustic waves within a medium. Typically, these waves are created within a duct or in our case the combustion chamber of the SRM. As the fluid propagates through the chamber, the sound produces changes with respect to any geometry changes and changes in the fluid's velocity. In the case of SRMs, three types of acoustic waves occur in the combustion chamber: longitudinal, tangential, and radial acoustic modes [5]. These modes are shown in Fig. 65.2. For this study, the longitudinal mode and its interaction with pressure oscillation is the key point of interest.

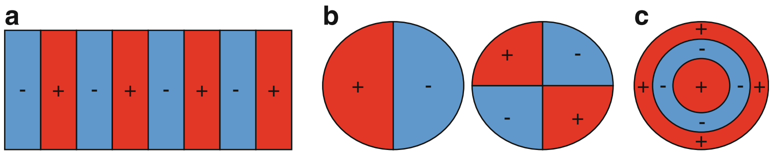
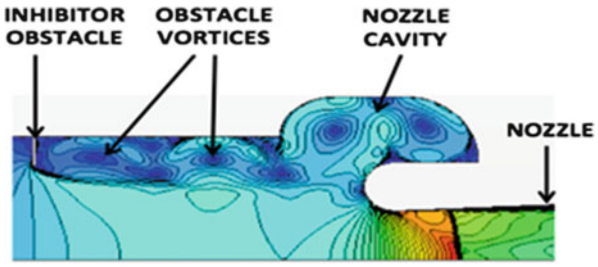


Fig. 65.2 Acoustic modes: (**a**) longitudinal, (**b**) tangential, and (**c**) radial

Fig. 65.3 Vortex shedding due to inhibitors



In the fluids subsystem, pressure oscillations are generated from turbulent flow within a rocket engine (e.g., from vortex shedding). There are three types of vortex shedding that can occur within an SRM: obstacle vortex shedding, surface vortex shedding, and parietal vortex shedding [6]. This paper will focus on obstacle vortex shedding as this was the major contributor to the T/O event on the Ares 1. Shearing layers that form around any blunt object that protrudes into the flow, the inhibitors in this case, generate obstacle vortex shedding. When these shear layers form, low-pressure vortices detach and flow downstream causing pressure oscillations [4] (Fig. 65.3).

Once pressure oscillationsare generated within the rocket chamber, the structure then responds to these cyclical loads by dampening out the excited energy in the system. The effectiveness of a structure to dampen out these cyclical loads is mostly determined by its geometry and material properties. If dampening overcomes the forces exerted onto it by pressure oscillation, the system will return to its stable state. However, in the instance of the Ares 1, excitation was larger than the dampening, leading to structural resonance of the system. These vibrations within the engine were not sufficiently dampened and propagated throughout the system, eventually affecting the crew module sufficiently as to impact the potential health and function of the crew.

65.2.3 *Multidisciplinary Design Optimization*

The design of complex engineered systems is ever more dependent on the ability to accurately capture and model interactions between disciplines. Typical engineering processes use a hierarchical decomposition of design and development tasks where coupling interactions are harder to capture. Unintended behaviors will often result

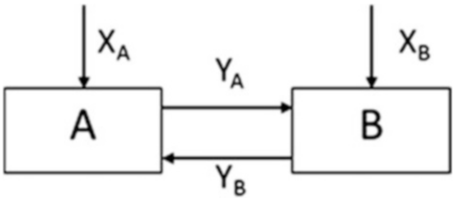


Fig. 65.4 Simple subsystem interaction

in complex systems if these interactions are not adequately modeled and represented during the design and development phase. Multidisciplinary design optimization (MDO) was developed in the early 1980s to specifically address interactions in large-scale complex engineering systems [7–9]. MDO research has focused on creating frameworks to enable accurate exploration of design spaces by capturing inherent couplings in both the physics and analysis of the system [10–13]. The process involves the creation of an objective function where the preference of system design is reflected, and the constraints represent design requirements.

To assess the impacts that design variables have on subsystem and system interactions, an iterative process is required to converge the system analysis. The process involves initializing system design variables and then iterating through a coupled analysis until convergence is met. A simple system is shown in Fig. 65.4, where design variables are denoted by (X_A,X_B) and the behavior variables are denoted by (Y_A,Y_B).

To determine overall system impact, sensitivities of subsystem couplings are analyzed through the implementation of a coupling strength analysis. The local and global derivatives are analyzed using the global sensitivity equation (GSE) method, which provides an efficient approach to obtain first-order sensitivity of the system's behavioral response with respect to design variables [7–9]. This is done by decomposing the larger system into smaller subsystems and evaluating the subsystem behavioral responses. For the two subsystem examples shown in Fig. 65.4, these system sensitivities are $\left(\frac{dY_A}{dX_A}, \frac{dY_A}{dX_B}, \frac{dY_B}{dX_A}, \frac{dY_B}{dX_B}\right)$, which are based on subsystem behavioral responses $\left(\frac{\partial Y_A}{\partial X_A}, \frac{\partial Y_A}{\partial Y_B}, \frac{\partial Y_B}{\partial X_A}, \frac{\partial Y_B}{\partial X_B}\right)$.This approach is aimed at solving the total derivative matrix of the system, which gives information on the influence of changes in one subsystem's output, due to changes in design variables within another subsystem. This interaction is shown below in Eq. 65.1 and Eq. 65.2, where the A matrix represents the sensitivity matrix and the total derivative matrix is solved by inverse matrix multiplication. The GSE for the two subsystem examples shown above is presented in Eqs. 65.1 and 65.2:

$$\begin{bmatrix} 1 & \frac{-\partial Y_A}{\partial Y_B} \\ \frac{-\partial Y_B}{\partial Y_A} & 1 \end{bmatrix} \begin{bmatrix} \frac{dY_A}{dX_A} & \frac{dY_A}{dX_B} \\ \frac{dY_B}{dX_A} & \frac{dY_B}{dX_B} \end{bmatrix} = \begin{bmatrix} \frac{\partial Y_A}{\partial X_A} & 0 \\ 0 & \frac{\partial Y_B}{\partial X_B} \end{bmatrix} \tag{65.1}$$

$$[A]\left[\frac{dY}{dX}\right] = \left[\frac{\partial Y}{\partial X}\right] \tag{65.2}$$

The advantage of using GSE is that the system needs to be converged only once at the beginning of the sensitivity analysis, which can dramatically save on computational time. These system sensitivities can then be used in a gradient-based optimization search, or can be used to determine relative importance of system couplings. By identifying the importance of system couplings, a comparative analysis can be made to identify those that critically impact system behaviors, which gives system managers a method to determine the importance of interactions that could otherwise be overlooked. To determine the value impact of these couplings, a system coupling approach is presented in the following section.

65.2.4 System Coupling Strength Approach

The ability to quantify the strength of coupling interactions within a systems context allows for a means to understand how these couplings impact the overall system value. Two methods in quantifying coupling strengths have been proposed: a local sensitivity-based approach and a system sensitivity-based approach [10, 13]. Previous research demonstrates that local sensitivities do not provide the overall systems impact toward value [10, 11]. System coupling strengths are developed to reflect the impact local couplings $[A_{ij}]$ have on the system requirements. This method focuses on the subsystem interactions. Once normalized, these sensitivities are used to determine which subsystem couplings have a sufficient impact on the system relative to the local design space. These papers demonstrate that following the comparative analysis, a determination is made into which couplings can be suspended or even removed in a total derivative analysis. The relevance for this work is that this analysis can be used to indicate which couplings are sufficiently significant that they should not be removed. In fact, if these couplings are not represented in the analysis, this could lead to unintended consequences, such as T/O.

65.2.5 Value-Driven Design

Value-driven design (VDD) is a proposed alternative to a traditional systems engineering approach [14]. VDD is designed to capture the true preferences of the stakeholders. This value is captured through a value function (objective function), which is a function of attributes, which in turn are functions of design variables. Additionally, VDD reduces the requirements; hence, a designer is able to explore the design space with more freedom [15]. The value is a singular unit, typically in the form of dollars for commercial enterprises, which correlates with the stakeholder's preference. This value enables direct comparisons to different design alternatives and aids in the decision making process [15]. In this paper, a value function is introduced as a method to capture the true preference of the stakeholders of Ares 1.

65.3 Developed Approach

Figure 65.5 presents a systems diagram that demonstrates the primary disciplines contributing to T/O, together with the behavior variables and design variables associated with the Ares I. The diagram in Fig. 65.5 is a design structure matrix (DSM) used here to represent the interrelations of disciplines. These subsystems have been shown to have a direct impact on T/O. The respective design variables were chosen from Blomshield's [18] work, as well as NASA data. These variables are identified in the nomenclature section of this paper. From the figure, we see that this is a coupled system requiring iteration between fluids, structures, and acoustics. The big question for this work is the degree to which each of the couplings impacts the T/O event and whether all need to be represented during design.

65.3.1 Metamodeling Pressure Oscillation

Initially, the T/O event was recreated using a CFD program (for the fluids) known as Star CCM+ and an FEA program (for structures) known as ANSYS WorkBench [13]. Star CCM+ was used to gather pressure oscillation data to create a forcing function that was then placed on the structures of the rocket in ANSYS. The

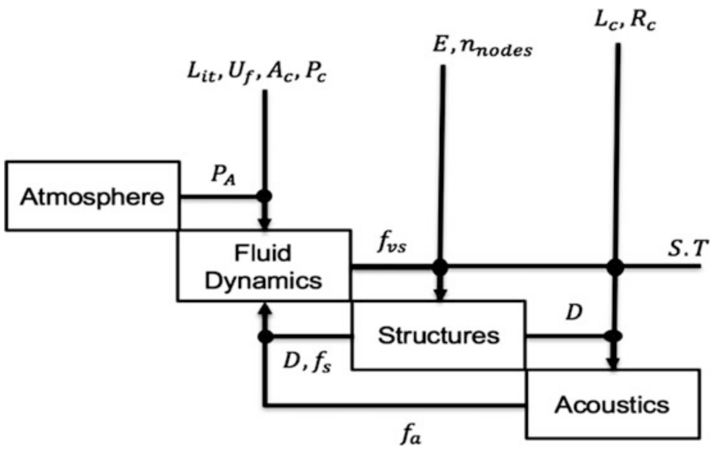


Fig. 65.5 System DSM

structure program would then calculate deflections and pass them back into the CFD program, thereby providing a high fidelity modeling approach. A lower fidelity modeling approach was developed, in which a metamodel was developed and used to determine the values of pressure along the walls of an SRM during a certain range of time [17]. The following equation was used to create the metamodel, where p_{ij} is the initial pressure on a two-dimensional grid, f_{ij} is the vortex shedding frequency, and t is the time:

$$p_i(t) = \sum_j p_{ij} \sin\left(2\pi f_{ij} t\right) + \mathrm{rand}([0:1])^* \left(P_{\mathrm{high}} - P_{\mathrm{low}}\right) \tag{65.3}$$

where

$$f_{ij} = \frac{S.T^* U_i}{L_i} \tag{65.4}$$

Vortex shedding frequency is determined using Eq. 65.3, where $S.T$ is a value that determines how oscillatory the flow is, L_i is the length of the obstacle protruding into the flow, and U_i is the velocity of the fluid [17]. To address uncertainties, random inputs were included to achieve results that were similar to those produced by Star CCM+. Star CCM+ was then used to initialize the vortex shedding frequency and initial pressures that occur within the SRM at a certain time t. The high fidelity fluids model was used to ensure that the quality of the fluids metamodel was sufficient for our analysis. The metamodel structure is shown in Fig. 65.6. Multiple analyses were performed to ensure that the metamodel output was matching the output of Star CCM+. The maximum error for pressure oscillations and frequencies was 3.8%, which was sufficient for the coupling strength analysis being performed.

65.3.2 *Coupling Between Acoustic and Fluids*

Transmissibility is used to capture the coupling interaction between acoustics and fluids, thereby establishing a relationship between the vortex shedding frequency and acoustic modes, which aids in determining the magnitude of the forces that are transmitted to the supporting structure [18]. The effect that this relationship has on

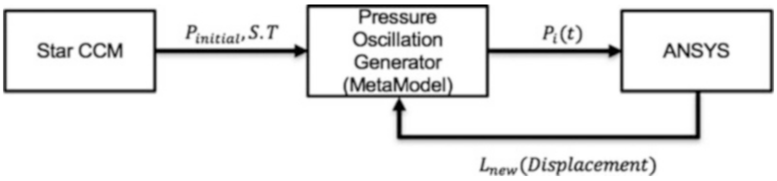


Fig. 65.6 Metamodel created for fluids

the magnitude of T/O is given by Eq. 65.5, where ζ represents damping, f_i is the vortex shedding frequency, f_n is the acoustic mode frequency, F_T is the transposed force, and F_O is the original force:

$$T = \frac{F_T}{F_O} = \frac{\sqrt{1 + \left(2\zeta \frac{f_i}{f_n}\right)^2}}{\sqrt{\left(1 - \left(\frac{f_i}{f_n}\right)^2\right)^2 + \left(\frac{2\zeta f_i}{f_n}\right)^2}} \tag{65.5}$$

65.3.3 Mathematical Model Developed to Define Magnitude of Thrust Oscillation

It is desirable to determine the impact that pressure oscillation has on the thrust produced by the solid rocket motor. A mathematical model was developed to determine the magnitude of thrust oscillation due to pressure oscillation. This model was developed from past research that approximated T/O due to multiple rocket instabilities [19]. In the past, it was assumed that to approximate T/O due to pressure oscillation, one needed to multiply the nozzle area by the peak-to-peak pressure oscillation [19]. This method yields smaller oscillations than what actually occurs inside the solid rocket motor. The following equations assume that longitudinal oscillations have a higher impact on T/O and the transverse modes cause no T/O [19]:

$$\Delta F = 2A_c \Delta P \tag{65.6}$$

$$\Delta F = \left[F + P_A A_E - \left(\bar{P}_N + \bar{P}_H\right) A_C\right] \frac{\Delta P}{\bar{P}_N} \tag{65.7}$$

The first equation is a simplified version that only takes into account the peak-to-peak pressure oscillations and the motor chamber area. The second equation incorporates greater fidelity, taking into account more design variables that impact T/O.

65.4 Systems Coupling Approach

Once the numerical methods and models were developed, a sensitivity analysis was performed to solve for the first-order subsystem interactions to determine important couplings. A similar approach was taken in [8, 13], in which local sensitivity analyses were performed using GSE. The matrix shown below has been normalized, enabling a valid comparison to evaluate which couplings have the greatest

Table 65.1 Local sensitivity results

Behavior variables (row and column)	f_{vs}	D_X	D_Y	D_Z	A_1	A_2	A_3
f_{vs}	1	−0.5E-3	−1.5E-3	0	0	0	0
D_X	9.993E-2	1	0	0	0	0	0
D_Y	−1.44E-2	0	1	0	0	0	0
D_Z	0	0	0	1	0	0	0
A_1	0	−1.13E-3	0	0	1	0	0
A_2	0	−1.85E-3	0	0	0	1	0
A_3	0	−1.39E-3	0	0	0	0	1

Table 65.2 Total derivative matrix results

Behavior variables (row) Design variable (column)	L_i	U_f	E	L_c
f_{vs}	−1.52E9	400.11	0	0
D_X	2.8	−7.37E-7	0	0
D_Y	20.178	−5.29E-6	1	0
D_Z	0	0	0	1
A_1	1981.1	−5.2E-4	0	−1.08E6
A_2	0	−1.6E-3	0	−4.2E6
A_3	0	−1.6E-3	0	−8.5E6

impacts. This shows a clear impact of fluids (through f_{vs}) on structures (D_X and D_Y) and a clear coupling between structures (D_X) and acoustics ($A_1 - A_3$), both of which were expected and modeled by the NASA design team. Surprisingly, there is also a coupling from structures (D_X andD_Y) back to fluids (f_{vs}). It is this coupling that quite possibly contributed to T/O and was originally thought to be insignificant (Table 65.1).

In addition to the local sensitivities, the global (total) system sensitivities were found. Table 65.2 shows that the length of the inhibitor,L_i, has a large impact on vortex shedding frequency,f_{vs}, and is a key contributor to T/O. This evaluation allowed a greater focus on the inhibitor and turbulence caused by object vortex shedding, as opposed to diversifying and evaluating all nodes within the structure which have a very weak association to vortex shedding in the system.

65.5 Ares 1 Value Model (Fig. 65.7)

This section focuses on the creation of a value model that will be used for a total coupling strengths analysis. A value function is used to aid in the decision-making process to reflect the true preference of the stakeholders. Here, the stakeholder considered is the administrator of NASA, whose role is to oversee the program and its success. Since the Ares 1 is a manned mission, there are many factors that affect

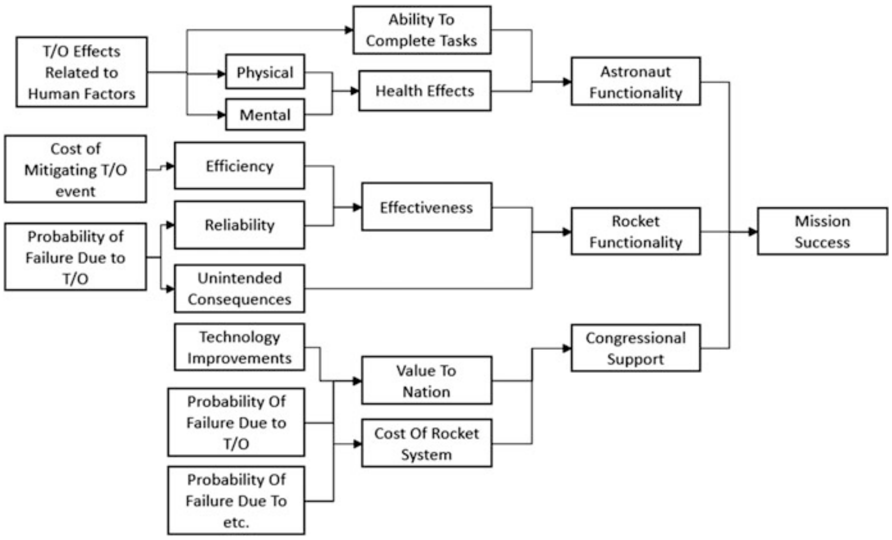


Fig. 65.7 Value model architecture

the overall mission success, including the astronaut's functionality, rocket functionality, and congressional support.

In modeling the astronauts' functionality, there are multiple attributes that must be captured in determining whether the astronauts can complete their tasks during an assigned mission. The attributes in this model are the astronauts' ability to complete tasks and any adverse long-term health effects [20]. An assumption is made that the magnitude of T/O will have a direct impact on the astronauts' ability to complete his or her tasks. This is separated into mental and physical backlash of T/O, where the magnitude of T/O impacts possible health effects that then contribute to mission success. The value of the system is then dependent upon the success rate of the SRM to maximize mission success, given the impacts of T/O captured in the local couplings analysis. For the congressional support model, Congress is treated as investors and each year their assets are invested into the newest space program. The Federal Government gives approximately 5% of its budget to NASA for research and development in space exploration [21]. If the proposed projects stay under their given budget, the government supports them 100%. However, in cases when projects go over budget, disincentives are added to account for loss of congressional support. This comes in the form of lower support for the project, which in turn provides less funding for NASA to complete its goals. However, any spin-offs created during the development process are considered to be revenue coming into the country, with their impact toward the country's gross national product, giving Congress an incentive to increase their investment [21].

65.6 Uncertainty

Uncertainty must be determined during the design and development of a system. It is important to capture uncertainty to predict how the system will behave while in operation. By predicting what is uncertain, analysis can be performed to determine if there should be changes to the design before it is complete. Essentially, the designer wants to predict the likelihood that known or unknown phenomena will occur during the system's operational lifecycle [19]. We know T/O has a high probability of occurring during an SRM's operation; however, it is hard to determine the magnitude of T/O that is occurring within the SRM. There are multiple behavior variables that cannot be predicted using mathematical equations, which need to be sampled multiple times to determine their value. An example is the Strouhal number. There are some constants for this value, but when analyzing the flow in a system, there are multiple phenomena that occur that can have a huge impact on the actual value. Hence, in this example, random samples are taken from a triangular distribution for each of the design variables and analyzed. Figure 65.8 shows the probability of the magnitude of T/O occurring within the SRM chamber (referring to Eqs. 65.6 and 65.7 as simple and complex). We see that variation in the Strouhal number can have a significant impact on the magnitude of T/O and probability of occurrence at that magnitude.

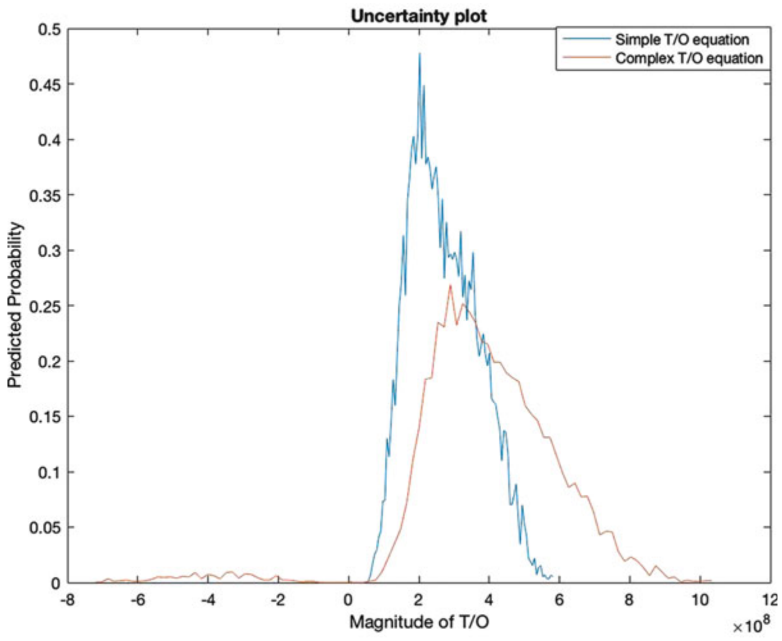


Fig. 65.8 Probability of T/O magnitude

65.7 Conclusion

This paper demonstrates that a critical coupling from the deformation of structural geometry on the vortex shedding frequency was determined to be a much more significant coupling than had originally been predicted and may have led to an increased probability of the T/O event. By capturing this feedback and associating a comparative value to its impact on system performance gives system managers a possible tool to capture unintended consequences during the design phase. Knowledge of the strength of this specific coupling between structures and fluids could have aided the Ares 1 design team in further research and inspection that might have eliminated the T/O event. Future work will investigate the impact of this coupling in a value model, where T/O is used as a means of evaluating the success rate of astronauts, rocket functionality, and congressional support on mission success. Further development is being made in the creation of the value functions shown in this paper. Prior data on pilot health during operation in fighter aircrafts is being considered to give more insight on variables that impact astronaut health. Once an accurate value model is developed, additional exploration on couplings will be implemented.

Acknowledgments The authors gratefully acknowledge support of this investigation by the System Engineering Consortium at University of Alabama, Huntsville (SUB 2015-034), sponsored by NASA (NNM11AA01A). We would also like to acknowledge Matthew Poetting, presently at Textron, who aided in the development of the high fidelity Star CCM+ model.

References

1. Constellation. Constellation finalizes thrust oscillation fix. 21 December 2009. Retrieved from https://blogs.nasa.gov/Constellation/2009/12/21/post_1261434125038/
2. Zhang Q, Wei Z, Su W, Li J, Wang N (2012) Theoretical modeling and numerical study for thrust-oscillation characteristics in solid rocket motors. J Propuls Power 28(2):312–322
3. Culick FEC (2002) Combustion instabilities in solid propellant rocket motors. Internal aerodynamics in solid rocket propulsion, Belgium
4. Guery J, et al (2008) Thrust oscillations in solid rocket motors.AIAA paper4789
5. Pirk R, d'Andrade Souto C, Donizeti da Silveira D, Mango de Souza C, Carlos Sandoval Goes L (2010) Liquid rocket combustion chamber acoustic characterization. J AerospTechnol Manag 2(3):269–278
6. Fabignon Y, Dupays J, Avalon G, Vuillot F, Lupoglazoff N, Casalis G, Prevost M (2003) Instabilities and pressure oscillations in solid rocket motors. Aerosp Sci Technol 7:191–200
7. Sobieszczanski-Sobieski J (1990) Sensitivity of complex, internally coupled systems. AIAA J 28(1):153–160
8. Hajela P, Bloebaum CL, Sobieszczanski-Sobieski J (1990) Application of global sensitivity equations in multidisciplinary aircraft synthesis. J Aircr 27(12):1002–1010
9. Sobieski J, Bloebaum CL, Hajela P (1991) Sensitivity of control-augmented structure obtained by a system decomposition method. AIAA J 29(2):264–270

10. English K, Miller E, Bloebaum C (1996) Total derivative-based coupling suspension for system reduction in complex design.6th symposium on multidisciplinary analysis and optimization
11. English K, Bloebaum CL, Miller E (2001) Development of multiple cycle coupling suspension in the optimization of complex systems. Struct Multidiscip Optim 22(4):268–283
12. English K, Bloebaum CL (2008) Visual dependency structure matrix for multidisciplinary design optimization Tradeoff studies. J Aerosp Comput Inf Commun 5(9):274–297
13. Bloebaum CL (1995) Coupling strength-based system reduction for complex engineering design. StructOptim 10:113–121
14. Collopy PD, Hollingsworth PM (2011) Value-driven design. J Aircr 48(3):749–759
15. Kannan H, Bloebaum CL,Memser BL (2015) Incorporation of coupling strength models in avalue-based systems engineering framework for optimization. AIAA Aviation 2015 (16th AIAA/ISSMO multidisciplinary analysis and optimization conference)
16. Kis D, Poetting M, Wenger C, Bloebaum CL (2016) A multidisciplinary coupling anlaysis mehod to support investigation of ares 1 thrust oscillation. CSER 2016 (Conference of Systems Engineer Research)
17. Mason DR, et al (2004) Pressure oscillations and structural vibrations in space shuttle RSRM and ETM-3motors.AIAA Paper3898:2004
18. Zhou Y-L, Maia NMM, Wahab MA (2016) Damage detection using transmissibility compressed by principal component analysis enhanced with distance measure. J Vib Control. 1077546316674544
19. Blomshield FS (2007) Lessons learned in solid rocket combustion instability. AIAA Paper5803:2007
20. Leveton LB, Stone L (1990) Crew selection, productivity and well-being for human exploration missions. No. 901362. SAE Technical Paper
21. Schnee J The economic impacts of the U.S. Space program.The economic impacts of the U.S. Space program. Business Administration Department Rutgers University, 1978. Web. 29 Oct 2016
22. Yao W et al (2011) Review of uncertainty-based multidisciplinary design optimization methods for aerospace vehicles. Prog Aerosp Sci 47(6):450–479

Chapter 66
Survey of Four Uncertainty Quantifications Methods in Systems Engineering

Ehsan Salimi, Andrea H. Cadenbach, and Ali E. Abbas

Abstract Uncertainty is an inherent property of engineering systems. Many methods for quantifying uncertainty have been proposed and applied to systems engineering. In this paper, we survey some of the literature on uncertainty quantification and comment on some criticisms of various approaches. We specifically focus on four selected methods: Monte Carlo simulation, cross entropy methods, fuzzy theory, and evidence theory.

Keywords Uncertainty quantification • Systems engineering • Probability

66.1 Introduction

Uncertainty is inherent in systems engineering, necessitating methods of uncertainty quantification to predict system performance and facilitate informed decision-making. Uncertainty quantification is a critical component to systems engineering, but in spite of this importance, disagreement persists how to quantify uncertainty and probability. For example, a purely frequentist approach assigns probability based on the ratio of successes to the total number of trials [1], while others emphasize that probability is a degree of belief [2]. Still others prefer to use Jaynes' maximum entropy principal and assign probabilities based on the available information [3, 4]. In many ways, these varied interpretations are facilitated by the work of Cox who shows that the laws of probability derive from a set of postulates and apply equally well regardless of the source of the initial probability assessments [5]. In the case that uncertainty about the initial assessments persists, Howard highlights that even this uncertainty can be modeled using classical probability theory [6]. The use of probability theory to measure uncertainty is further supported by the work of Lindley who shows that every reasonable scoring function

E. Salimi • A.E. Abbas (✉)
Industrial and Systems Engineering, University of Southern California, Los Angeles, CA, USA
e-mail: aliabbas@usc.edu

A.H. Cadenbach
Logistics and Operations Management, University of Missouri-St. Louis, St. Louis, MO, USA

A.M. Madni et al. (eds.), *Disciplinary Convergence in Systems Engineering Research*, https://doi.org/10.1007/978-3-319-62217-0_66

describing uncertainty should rely on probability; failing to use probability results in an inconsistency [7, 8].

More recently, the emergence of artificial intelligence has stimulated the establishment of new approaches to describing uncertainty. Many of these methods are inspired by the nonexact nature of linguistic communication [9]. For example, words such as “tall” or “much” give qualitative, inexact information that leaves uncertainty in quantitative, exact measure. Fuzzy theory [10] and evidence theory [11, 12] are among the most well-known methods based on this general principal.

The variety of approaches to uncertainty quantification means that the analyst of a systems engineering application must select the approach(es) best suited to the particular problem of interest. But the differing perspectives on uncertainty confound this selection. In this paper, we address the issue of selecting the best method to quantify uncertainty by reviewing the literature on both probabilistic and nonprobabilistic methods to provide insight to the contrasts between the views. Specifically, the probabilistic methods reviewed are Monte Carlo simulation (MCS) and entropy methods, while the non-probabilistic methods are fuzzy probability and evidence theory. The paper is organized around two primary goals:

1. Providing a brief review of four common approaches to uncertainty quantification and
2. Discussing the applications of each method in systems engineering

We begin the survey in Sect. 66.2 with some additional background on the different types of uncertainty quantification methods. Section 66.3 examines two probabilistic methods: MCS and entropy methods. Section 66.4 examines two nonprobabilistic methods: fuzzy probability and evidence theory. Finally, Sect. 66.5 concludes the paper.

66.2 Classifying Uncertainty Quantification Methods

The emergence of uncertainty quantification methods in artificial intelligence creates a split in perspectives on uncertainty quantification methods. The traditional view holds that probability theory is the only logical approach to uncertainty quantification, and probabilistic methods should be used. This view is agnostic on the source of probability measures, treating degrees of belief and reasonable expectations about an event as though they are indeed the probability of that event occurring. However, in the artificial intelligence community, some argue that uncertainty is not necessarily the same as randomness and that it is necessary to distinguish randomness and ambiguity as two different concepts of uncertainty [13].

The most common view on classifying uncertainty is to divide it based on the origin of the uncertainty [13, 14]. This view classifies uncertainty as either reducible or irreducible. Irreducible uncertainty, also known as aleatory uncertainty, is discussed as being the inherent and natural variability of the physical system under study. Based on this view, irreducible uncertainty is the only type of uncertainty that can be described by classical probability theory. Reducible uncertainty or epistemic uncertainty, on the other hand, is described as the lack of certainty due to the shortcomings of scientific models. Limited knowledge or imprecise data are introduced as the main cause of such uncertainty in a system. It is argued that the ambiguity in the system is deterministic in nature; hence, probability theory cannot capture it. This difference in describing uncertainty leads to the two different perspectives in uncertainty quantification, the probabilistic view and nonprobabilistic view.

Before reviewing uncertainty quantification methods of each viewpoint, it is important to consider that the application of any method must be appropriate for the type of problem at hand. Problems involving uncertainty can be classified as one of two types: the forward and inverse problem. The forward problem deals with uncertainty propagation in a system. The inverse problem estimates the parameters of the proposed model based on the observed/measured noisy data. The inverse problem, unlike the forward problem, is an ill-posed problem as the existence of the solution or uniqueness of the solution, if exits, is not guaranteed [15]. The presence of high-dimensional data can make both problems computationally intractable [16]. The discussion about computational complexity of these methods is beyond the scope of this paper. However, we mention that there are few approaches such as model reduction [17] and spectral methods [18] to overcome this issue.

Both probabilistic and nonprobabilistic methods have been applied to forward and inverse problems. In the following sections, this paper reviews a method from each perspective for each type of problem.

66.3 Probabilistic Methods

66.3.1 Monte Carlo Simulation

In its simplest form, Monte Carlo simulation (MCS) is a sampling method which captures the distribution of the output of the system by sampling from its input. By its nature, MCS addresses the forward problem of uncertainty propagation in a system. To formally describe the method, let X represent the input for the system and Y represent the outcome of the system. Both X and Y can be considered multidimensional vectors. The relationship between the input and output can be simplified by a function/simulator f, where:$Y = f(x)$. The MCS method draws n points randomly from the distribution of $X, \{x_i : i = 1, 2, \ldots, n\}$, which is assumed to be known. Using the simulator/function f, n samples from the output distribution is obtained:

$$\{y_i = f(x_i) : \quad i = 1, 2, \ldots, n\} \tag{66.1}$$

By considering the relative proportion/volume of each outcome, the previously unknown output distribution can be characterized. The law of large numbers (LLN) guarantees that the estimation of the output distribution found using MCS will eventually converge to the true distribution of the outcome as the sample size increases toward infinity.

Although MCS appears simple to implement, care must be exercised in designing the underlying simulation to ensure the method produces meaningful results. Different sampling methods, such as choosing to sample with or without replacement, can lead to different results [19, 20]. In addition, the sample size should be sufficiently large to reduce error in the estimation; otherwise, the results can be faulty [21]. However, a large sample size can add significant computational burden and more complexity to the system if there are unknown parameters in the description of the underlying system [22].

Advances in computers and computing platforms have made it easier to utilize MCS to capture the uncertainty propagation both in scientific and industrial applications of high complexity. Sometimes it seems the only feasible option to model a complex system [23]. It is, therefore, widely applied in systems engineering contexts [24, 25].

Advances in computing have also made MCS an attractive method for applications in many other engineering and scientific fields in which complex systems are present. MCS is an essential tool in financial engineering for derivative pricing and risk management [26]. It has widespread application in risk analysis and decision-making under uncertainty [27, 28] as well as stochastic optimization [29–32]. The application extends beyond the engineering systems to fields such as computational physics [33, 34] and computational biology [35, 36], among others.

66.3.2 Entropy Methods

Entropy methods are based on Shannon's description of "entropy" as a measure of choice involved in the outcome of a process or as the measure of how "uncertain" we are about the outcome [37]. In his seminal work, he argues that the measure is reasonable for a discrete random variable if it satisfies certain properties such as continuity in probability and reaching a maximum under equiprobability. He proves the only measure satisfying the above properties for a discrete random variable X is:

$$(x) = -\sum_{i=1}^{n} p(x_i)\log p(x_i) \tag{66.2}$$

where $p(x_i)$ represents the probability of outcome x_i, for $i = 1, \ldots, n$.

This definition of entropy has become a widely applied measure of uncertainty in numerous fields. In the areas of communications engineering, information theory, and cryptography, it is an essential measure of information and uncertainty

[38]. Pun uses it to define the entropy of an image [39, 40], facilitating the application of Shannon's entropy in image processing [41–43]. In the area of control theory, entropy and information theoretic results are used in control problems in the context of networks with communication [44–46].

Additional applications of entropy are made possible by Renyi's generalization of Shannon's entropy such that it satisfies additivity [47]. The Renyi generalization has found many applications in communication security [48, 49]. Massey [50], Arikan [51], and others [52, 53] apply the Renyi generalized entropy in making tighter bounds in guessing problems. Nonlinear dynamic systems [54, 55] and pattern recognition and machine learning [56, 57] are among the other applications of Renyi entropy.

Entropy can also be used to assign probability distributions over outcomes based on the amount of available information. These assignments are useful in inverse problems of uncertainty quantification and are based on the principles of maximum entropy and of minimum cross entropy. Jaynes introduced the principle of maximum entropy [58]. This principle argues that among all the distributions P which satisfy a given set of constraints, one should choose a distribution, P^* with the largest entropy or highest uncertainty. These set of constraints represents the partial information available to experts. Using notation introduced previously, the principal of maximum entropy can be posed as following optimization problem:

$$P^* : \text{argmax} - \sum_{i=1}^{n} p(x_i)\log p(x_i) \tag{66.3}$$

$$\sum_i p(x_i) = 1$$

$$E\left[f_j(x_i)\right] = \mu_j, \quad j \in J$$

$$p(x_i) \geq 0$$

This is a specific formulation of the maximum entropy model where the set of available information, $E[f_j(x_i)] = \mu_j$ is represented by moment functions f_j.

The principle of minimum cross entropy is introduced as a general framework for approximating the probability distribution with partial information [59]. The principle is based on minimizing the so-called Kullback–Leibler divergence [60] which is defined as:

$$K(P:Q) = \sum_{i=1}^{n} p(x_i)\log\frac{p(x_i)}{q(x_i)} \tag{66.4}$$

In the expression above, distribution P is the reference distribution estimated with distribution Q, where $p(x_i)$ and $q(x_i)$ represent the probabilities for outcome $i = 1, \ldots, n$, of distributions P and Q, respectively. The Kullback–Leibler divergence measures the amount of information lost when the underlying distribution P is estimated with Q. If the prior distribution is a uniform distribution, then the minimum cross entropy distribution becomes the same as maximum entropy distribution.

The application of the principles of maximum entropy and of minimum cross entropy is bolstered by the results of Shore and Johnson [61]. They prove that these principles are correct methods to be used in inference problems with partial information, where the information is presented as expected values.

The strong theoretical foundation for entropy optimization methods makes them attractive for applications in engineering and science [62, 63]. The methods have been used extensively in statistical mechanics and thermodynamics [64–66], leading to additional applications in measuring uncertainty of physical quantities [67, 68]. Berger et al. use the maximum entropy method in natural language processing [69]. Many applications related to machine learning also rely on maximum entropy [70–72]. Designing robust controllers also heavily utilizes the notion of entropy optimization [73–76]. Additional notable application areas include reliability engineering [77], traffic networks [78], system analysis [79], and financial engineering and stock market price distribution [80], among many others.

New applications of entropy optimization continue to be identified. For example, Shore and Gary have recently shown an application to pattern recognition [81]. Abbas shows applications in the field of decision-making under uncertainty. He discusses this method in approximating the underlying joint probability distribution for the decision alternatives in various scenarios [82–87]. He also uses the concept in the problem of aggregating experts' opinions to elicit the underlying probability distribution [88].

66.4 Nonprobabilistic Methods

This paper includes nonprobabilistic methods to provide perspective on the opposing viewpoints of uncertainty quantification as part of the survey. The inclusion of nonprobabilistic methods should not be interpreted as advocacy for their use.

66.4.1 Fuzzy Probability

Fuzzy theory is a concept introduced by Zadeh [10] that has evolved to be a new and distinct paradigm for quantifying uncertainty. To put it simply, fuzzy theory deals with sets that have no sharp boundaries. Examples of such a system can be the set of "tall men" or the set of numbers "much larger than 100." This theory contrasts with classic logic and probability theory where the membership to a set is either 1 or zero such that the defined boundaries are sharp.

The argument for fuzzy theory is that there is a difference between the fuzziness and the randomness in a system. The fuzziness arises when there are no sharp boundaries to distinguish the membership and nonmembership of an object in a set. On the other hand, the randomness is concerned with the membership problem in a sharp defined set (nonfuzzy set). Mathematically, for a collection of objects $X = \{x\}$, the fuzzy set A in X is defined as a set of ordered pairs:

$$A = \{(x, \mu_A(x)) | x \in X\}$$

The term $\mu_A(x)$ is called the "grade of membership" of x in A. In classic logic, this number is either 0 or 1, whereas in fuzzy theory, it belongs to the interval [0, 1]. Additional detail is available in the studies by Zadeh [10] and Zimmermann [89].

Fuzzy theory has elicited a great deal of discussion on both its merits and its faults. Hisdal [90] argues that fuzziness is a relative concept; it can be resolved by defining exact thresholds or exact conditions of observation. However, to illustrate the problem of clarity and inconsistency in fuzzy theory, consider preference modeling in a fuzzy setting. The ranking of alternatives in a fuzzy setting is done by a preference degree function: $\mu_p(A, B) = p_{AB}$. This value is between 0 and 1. The preference matrix can then be constructed between n alternatives, where $p_{AB} + p_{BA} = 1$. It is then easy to see that transitivity can be violated in this relation, meaning that if someone slightly prefers A to B and B to C, then he or she can prefer C to A. From a rational or normative perspective, such intransitivity violates rules of logic and is problematic. Proponents of fuzzy logic, however, argue that human preference functions are not exact and may lead to inconsistencies in practice, arguing that such intransitivity is not problematic. The topic of intransitivity with fuzzy logic has garnered a great deal of attention in the literature [91–94].

An additional criticism of fuzzy logic comes from Lindley [7, 8]. He shows that assignments of possibility based on fuzzy logic violate the additivity assumption of a scoring rule. Hence, no scoring rule for uncertainty leads to fuzzy logic; if one accepts the tenets of scoring rules, one cannot use fuzzy logic.

In spite of the lively debate on the properties of fuzzy logic, it has nonetheless been applied in the context of decision-making [95, 96]. The notion of a decision or a "fuzzy decision" is the process of choosing a set of choices among available alternatives. In contrast to classical decision theory, fuzzy decisions consist of goals, a constraint set, and alternatives which are all fuzzy sets. Fuzzy theory has also been widely applied to engineering systems in areas such as control theory, pattern recognition, and natural language processing [97–99]. A growing body of research is considering the application of fuzzy theory in engineering design, specifically dealing with imprecisions [100–102].

66.4.2 Evidence Theory

Evidence theory is a concept introduced by Dempster [11, 12] and later developed by Shafer [103]. Also known as the "Dempster–Shafer Theory of Belief," it is an attempt to model what is called "*epistemic uncertainty,*" based on an argument that classic probability theory does not assign any probability distribution over such an uncertainty. The Dempster–Shafer theory assumes that the realization of such an uncertainty is obtainable by sampling methods. The development of evidence theory reflects a desire from the artificial intelligence community to combine traditional rigorous probability theory with a more flexible rule-based system. As Shafer states [104, 105], evidence theory is based on two ideas: (1) assigning a degree of belief for one event/question based on the subjective probabilities that exist for related events/questions and (2) Dempster's rule of aggregation: combining degrees of belief for two independent sets of belief assignments. Formally, let set $X = \{x\}$ be the set representing all possible outcomes of a system. The basic belief assignment (BBA) or basic probability assignment (BPA) assigns a number between zero and one for each subset of X (remember that there will $2^{|X|}$ such subsets):

$$m : 2^{|X|} \to [0, 1]$$

where $\sum_{s \subset X} m(s) = 1$. The value of $m(s)$ is the proportion of all possible evidence that supports that an event belong to set s. Now, we can define an interval for the probability of a subset s with two new measures, belief and plausibility:

$$Bel(s) = \sum_{A \subseteq s} m(A)$$

$$Pl(s) = \sum_{A \cap s \neq \varnothing} m(A)$$

The interval then is defined as: $Bel(s) \leq P(s) \leq Pl(s)$. Note that

$$Pl(s) + Pl(s^c) \geq 1$$

$$Bel(s) + Bel(s^c) \leq 1$$

Guan and Bell [106] present a thorough list of applications for Dempster–Shafer theory. The artificial intelligence community heavily uses this method for classification and pattern recognition. Bloch [107], Denoeux [108–111], and others [112] use the Dempster–Schafer theory for tackling classification problems. The application of this theory is also discussed in the area of decision-making under uncertainty [113, 114], distributed decision-making [115], expert systems [116], engineering design [101, 117], engineering optimization [118, 119], and reliability engineering [120, 121].

66.5 Conclusion

In this paper, we review four common methods deriving from two different perspectives on uncertainty quantification: Monte Carlo simulation, entropy methods, fuzzy theory, and evidence theory. These methods are used for both the forward and inverse problems. We highlight applications of these methods in systems engineering.

We show that probabilistic methods have a solid theoretical foundation, while nonprobabilistic methods have been the subject of criticism, particularly in applications to decision-making. In spite of such criticism, applications in the artificial intelligence community persist.

Acknowledgements This work was supported by the National Science Foundation awards CMMI 15-65168, CMMI 16-29752 and CMMI 16-44991.

References

1. Bergmann G (1940) The logic of probability. Am J Phys 9(5):263–272
2. Ramsey FP (1931) Truth and probability (1926). In: The foundations of mathematics and other logical essays, pp 156–198
3. Jaynes ET (2003) Probability theory: the logic of science. Cambridge University press, Cambridge
4. Jaynes ET (1982) On the rationale of maximum-entropy methods. Proc IEEE 70(9):939–952
5. Cox RT (1946) Probability, frequency and reasonable expectation. Am J Phys 14:1–13
6. Howard RA (1988) Uncertainty about probability: a decision analysis perspective. Risk Anal 8(1):91–98
7. Lindley DV (1982) Scoring rules and the inevitability of probability. Int Stat Rev 50(1):1–11
8. Lindley DV (1987) The probability approach to the treatment of uncertainty in artificial intelligence and expert systems. Stat Sci 2(1):17–24
9. Zadeh LA (2005) Toward a generalized theory of uncertainty (GTU). Inf Sci 172(1):1–40
10. Zadeh LA (1965) Fuzzy sets. Inf Control 8(3):338–353
11. Dempster AP (1967) Upper and lower probabilities induced by a multivalued mapping. Ann Math Stat 38(2):325–339
12. Dempster AP (1968) A generalization of Bayesian inference. J R Stat Soc 30:205–247
13. Oberkampf WL, Diegert KV, Alvin KF, Rutherford BM (1998)Variability, uncertainty, and error in computational simulation. ASME-Publication HTD
14. Oberkampf WL, Helton JC, Sen K (2001) Mathematical representation of uncertainty, pp 16–19
15. Tarantola A (2005) Inverse problem theory and methods for model parameter estimation. Siam
16. Kaipio J, Somersalo E (2006) Statistical and computational inverse problems. Springer Science & Business Media, New York
17. Kaipio J, Somersalo E (2007) Statistical inverse problems: discretization, model reduction and inverse crimes. J Comput Appl Math 198(2):493–504
18. Marzouk YM, Najm HN, Rahn LA (2007) Stochastic spectral methods for efficient Bayesian solution of inverse problems. J Comput Phys 224(2):560–586
19. Sawilowsky SS (2003) You think you've got trivials? J Mod App Stat Meth 2(1):218–225

20. Sawilowsky SS, Fahoome GC (2003) Statistics via Monte Carlo simulation with Fortran. JMASM, Rochester Hills
21. Banks J (1998) Handbook of simulation: principles, methodology, advances, applications, and practice. John Wiley & Sons, New York
22. Liu JS (2008) Monte Carlo strategies in scientific computing. Springer Science & Business Media, New York
23. Kroese DP, Taimre T, Botev ZI (2013) Handbook of Monte Carlo methods. John Wiley & Sons, New York
24. Dubi A (2000) Monte Carlo applications in systems engineering. Wiley, Chichester
25. Rubinstein RY, Kroese DP (2011) Simulation and the Monte Carlo. John Wiley & Sons, New York
26. Glasserman P (2003) Monte Carlo methods in financial engineering. Springer Science & Business Media, New York
27. Jackel P, 2001 Monte Carlo methods in finance, stochastic dynamics
28. Mun J (2006) Modeling risk: applying Monte Carlo simulation, real options analysis, forecasting, and optimization techniques. John Wiley & Sons, Hoboken
29. Fu MC (2002) Optimization for simulation: theory vs. practice. INFORMS J Comput 14 (3):192–215
30. Robbins H, Monro S (1951) A stochastic approximation method. Ann Math Statistics 22 (3):400–407
31. Marinari E, Parisi G (1992) Simulated tempering: a new Monte Carlo scheme. Europhys Lett 19(6):451
32. Shapiro A (1996) Simulation based optimization. In: Proceedings of the 28th winter simulation conference
33. Rosenbluth MN, Rosenbluth AW (1955) Monte Carlo calculation of the average extension of molecular chains. J Chem Phys 23(2):356–359
34. Allen MP, Tildesley DJ (1989) Computer simulation of liquids. Oxford university press, New York
35. Milik M, Skolnick J (1993) Insertion of peptide chains into lipid membranes: an off-lattice Monte Carlo dynamics model. Proteins: Structure, Function, and Bioinformatics 15(1):10–25
36. Ojeda P, Garcia ME, Londoño A, Chen N-Y (2009) Monte Carlo simulations of proteins in cages: influence of confinement on the stability of intermediate states. Biophysical Journal 96 (3):1076–1082
37. Shannon C (1949) Communication theory of secrecy systems. Bell Syst Tech J 28:656–715
38. Cover TM, Thomas JA (2012) Elements of information theory. John Wiley & Sons, New York
39. Pun T (1980) A new method for grey-level picture thresholding using the entropy of the histogram. Signal Process 2(3):223–237
40. Pun T (1981) Entropic thresholding, a new approach. Computer Graphics and Image Processing 16(3):210–239
41. Pal SK (1982) A note on the quantitative measure of image enhancement through fuzziness. IEEE Trans Pattern Anal Mach Intell 4(2):204–208
42. Kapur JN, Sahoo PK, Wong AK (1985) A new method for gray-level picture thresholding using the entropy of the histogram. Comput Vis Graph Image Process 29(3):273–285
43. Pal NR, Pal SK (1991) Entropy: a new definition and its applications. IEEE Trans Syst Man Cybern 21(5):1260–1270
44. Ho Y-C, Kastner MP, Wong E (1978) Teams, signaling, and information theory. IEEE Trans Autom Control 23(2):305
45. Tatikonda S, Mitter S (2004) Control under communication constraints. IEEE Trans Autom Control 49(7):105–1068
46. Franceschetti M, Minero P (2014) Elements of information theory for networked control systems. In: Information and control in networks, Springer International Publishing, pp 3–37

47. Renyi A (1961) On measures of entropy and information. In: In Fourth Berkeley symposium on mathematical statistics and probability
48. Bennett CH, Brassard G, Crépeau C, Maurer UM (1995) Generalized privacy amplification. IEEE Trans Inf Theory 41(6):1915–1923
49. Hayashi M (2011) Exponential decreasing rate of leaked information in universal random privacy amplification. IEEE Trans Inf Theory 57(6):3989–4001
50. Massey JL (1994) Guessing and entropy. In: IEEE international symposium on information theory
51. Arikan E (1996) An inequality on guessing and its application to sequential decoding. IEEE Trans Inf Theory 42(1):99–105
52. Hanawal MK, Sundaresan R (2011) Guessing revisited: a large deviations approach. IEEE Trans Inf Theory 57(1):70–778
53. Christiansen MM, Duffy KR (2013) Guesswork, large deviations, and Shannon entropy. IEEE Trans Inf Theory 59(2):796–802
54. Erdogmus D, Principe JC (2002) An error-entropy minimization algorithm for supervised training of nonlinear adaptive systems. IEEE Trans Signal Process 50(7):1780–1786
55. Erdogmus D, Principe JC (2002) Generalized information potential criterion for adaptive system training. IEEE Trans Neural Netw 13(5):1035–1044
56. Gokcay E, Principe JC (2002) Information theoretic clustering. IEEE Trans Pattern Anal Mach Intell 24(2):158–171
57. Hild K, Erdogmus D, Torkkola K, Principe JC (2006) Feature extraction using information-theoretic learning. IEEE Trans Pattern Anal Mach Intell 28(9):1385–1392
58. Jaynes E (1957) Information theory and statistical mechanics. Phys Rev 106:620
59. Kullback S (1997) Information theory and statistics. Courier corporation, New York
60. Kullback S, Leibler R (1951) On information and sufficiency. Ann Math Stat 22:79–86
61. Shore J, Johnson R (1980) Axiomatic derivation of the principle of maximum entropy and the principle of minimum cross-entropy. IEEE Trans Inf Theory 26(1):26–37
62. Kapur JN (1989) Maximum-entropy models in science and engineering. John Wiley & Sons, New York
63. Kapur JN, Kesavan HK (1992) Entropy optimization principles and their applications. In: Entropy and energy dissipation in water resources, Springer Netherlands, pp 3–20
64. de Beauregard OC, Tribus M (1990) Information theory and thermodynamics. In: Maxwell's Demon: entropy, information, computing, Princeton University Press
65. Tribus M (1960) Thermostatics and thermodynamics. Center for Advanced Engineering Study, Massachusetts Institute of Technology, Cambridge
66. Tribus M (1961) Information theory as the basis for thermostatics and thermodynamics. J Appl Mech 28(1):1–8
67. Woeger W (1987) Probability assignment to systematic deviations by the principle of maximum entropy. IEEE Trans Instrum Meas 1001(2):655–658
68. Iuculano G, Nielsen L, Zanobini A, Pellegrini G (2007) The principle of maximum entropy applied in the evaluation of the measurement uncertainty. IEEE Trans Instrum Meas 56 (3):717–722
69. Berger AL, Della Pietra VJ, Della Pietra SA (1996) A maximum entropy approach to natural language processing. Comput Linguist 22(1):39–71
70. Della Pietra S, Della Pietra V, Lafferty J (1997) Inducing features of random fields. IEEE Trans Pattern Anal Mach Intell 19(4):380–393
71. Och FJ, Ney H (2002) Discriminative training and maximum entropy models for statistical machine translation. In: Proceedings of the 40th annual meeting on association for computational linguistics
72. Borthwick A (1999) A maximum entropy approach to named entity recognition, Doctoral Dissertation, New York University

73. Hyland DC (1984) Application of the maximum entropy/optimal projection control design approach for large space structures, NASA. Langley Research Center Large Space Antenna Systems Technol
74. Bernstein DS, Hyland DC (1985) The optimal projection/maximum entropy approach to designing low-order, robust controllers for flexible structures. In: 24th IEEE conference on decision and control
75. Dennis S (1986) Robust controller synthesis using the maximum entropy design equations. IEEE Trans Autom Control 31(4):362–364
76. Bernstein DS, Hyland DC (1988) Optimal projection for uncertain systems (OPUS): a unified theory of reduced-order, robust control design. Large space structures: dynamics and control. In: Large space structures: dynamics and control. Springer, Berlin Heidelberg, pp 263–302
77. Tribus M (1962) The use of the maximum entropy estimate in the estimation of reliability. Macmillan, New York
78. Beneš VE (1965) Mathematical theory of connecting networks and telephone traffic. Academic press, San Diego
79. Ferdinand AE (1970) A statistical mechanical approach to systems analysis. IBM J Res Dev 14(5):539–547
80. Cozzolino JM, Zahner MJ (1973) The maximum-entropy distribution of the future market price of a stock. Oper Res 21(6):1200–1211
81. Shore JE, Gray RM (1982) Minimum cross-entropy pattern classification and cluster analysis. IEEE Trans Pattern Anal Mach Intell 4(1):11–17
82. Abbas AE (2003) An entropy approach for utility assignment in decision analysis. In: Bayesian inference and maximum entropy methods in science and engineering, AIP conference proceeding
83. Abbas AE (2003).Entropy methods for univariate distributions in decision analysis. In: Bayesian inference and maximum entropy methods in science and engineering
84. Abbas AE (2006) Entropy methods for joint distributions in decision analysis. IEEE Trans Eng Manag 53(1):146–159
85. Abbas AE (2005).Maximum entropy distributions between upper and lower bounds. In: Bayesian inference and maximum entropy methods in science and engineering
86. Abbas AE, Aczél J (2010) The role of some functional equations in decision analysis. Decis Anal 7(2):215–228
87. Salimi E, Abbas AE (In press) A simulation-based comparison of maximum entropy and copula methods for capturing non-linear probability dependence. In: Winter simulation conference
88. Abbas AE (2009) A Kullback-Leibler view of linear and log-linear pools. Decis Anal 6 (1):25–37
89. Zimmermann HJ (2010) Fuzzy set theory. Wiley Interdiscip Rev Comput Stat 2(3):317–332
90. Hisdal E (1988) Are grades of membership probabilities? Fuzzy Sets Syst 25(3):325–348
91. Basile L (1990) Ranking alternatives by weak transitivity relations. In: Multiperson decision making models using fuzzy sets and possibility theory, Springer Netherland, pp 105–112
92. Tanino T (1984) Fuzzy preference orderings in group decision making. Fuzzy Sets Syst 12 (2):117–131
93. Herrera-Viedma E, Herrera F, Chiclana F, Luque M (2004) Some issues on consistency of fuzzy preference relations. Eur J Oper Res 154(1):98–109
94. Switalski Z (2003) General transitivity conditions for fuzzy reciprocal preference matrices. Fuzzy Sets Syst 137:85–100
95. Bellman RE, Zadeh LA (1970) Decision-making in a fuzzy environment. Manag Sci 17:141–164
96. Zadeh LA (1973) Outline of a new approach to the analysis of complex systems and decision processes. IEEE Trans Syst Man Cybern 3(1):28–44
97. Zimmermann HJ (1996) Fuzzy control. In: Fuzzy set theory—and its applications. Springer Netherlands, Dordrecht, pp 203–240

98. Dubois DJ (1980) Fuzzy sets and systems: theory and applications. Academic press, New York
99. Ross TJ (2009) Fuzzy logic with engineering applications. John Wiley & Sons, New York
100. Wood KL, Otto KN, Antonsson EK (1992) Engineering design calculations with fuzzy parameters. Fuzzy Sets Syst 52(1):1–20
101. Antonsson EK, Otto KN (1995) Imprecision in engineering design. J Vib Acoust 117 (B):25–32
102. Rao SS, Rao SS (2009) Engineering optimization: theory and practice. John Wiley & Sons, Hoboken
103. Shafer GA (1976) A mathematical theory of evidence. Princeton University press, Princeton
104. Shafer G (1990) Perspectives on the theory and practice of belief functions. Int J Approx Reason 4:323–362
105. Shafer G, Pearl J (1990) Readings in uncertain reasoning. Morgan Kaufmann Publishers Inc., San Francisco
106. Guan JW, Bell DA (1991) Evidence theory and its applications. Elsevier Science Inc, New York
107. Bloch I (1996) Some aspects of Dempster-Shafer evidence theory for classification of multi-modality medical images taking partial volume effect into account. Pattern Recogn Lett 17 (8):905–915
108. Denoeux T (1997) Analysis of evidence-theoretic decision rules for pattern classification. Pattern Recogn 30(7):1095–1107
109. Denoeux T (1995) A k-nearest neighbor classification rule based on Dempster-Shafer theory. IEEE Trans Syst Man Cybern 25(5):804–8013
110. Denoeux T (1999) A neural network classifier based on Dempster-Shafer theory. IEEE Trans Syst Man Cybern Part A Syst Hum 30(2):131–150
111. Denoeux T (1999) Reasoning with imprecise belief structures. Int J Approx Reason 20 (1):79–111
112. Horiuchi T (1998) Decision rule for pattern classification by integrating interval feature values. IEEE Trans Pattern Anal Mach Intell 20(4):440–448
113. Yager RR (1992) Decision making under Dempster-Shafer uncertainties. Int J Gen Syst 20 (3):233–245
114. Yang J-B, Singh MG (1994) An evidential reasoning approach for multiple-attribute decision making with uncertainty. IEEE Trans Syst Man Cybern 24(1):1–18
115. Drakopoulos E, Lee CC (1992) Decision rules for distributed decision networks with uncertainties. IEEE Trans Autom Control 37(1):5–14
116. Beynon M, Cosker D, Marshall D (2001) An expert system for multi-criteria decision making using Dempster Shafer theory. Expert Syst Appl 20(4):357–367
117. Butler AC, Sadeghi F, Rao SS, LeClair SR (1995) Computer-aided design/engineering of bearing systems using the Dempster-Shafer theory. Artif Intell Eng Des Anal Manuf 9 (01):1–11
118. Chen L, Rao SS (1998) A modified Dempster-Shafer theory for multicriteria optimization. Eng Optim 30(3–4):177–201
119. Yang J-B, Sen P (1997) Multiple attribute design evaluation of complex engineering products using the evidential reasoning approach. J Eng Des 8(3):211–230
120. Tanaka K, Klir GJ (1999) A design condition for incorporating human judgement into monitoring systems. Reliab Eng Sys Saf 65(3):251–258
121. Simon C, Weber P, Evsukoff A (2008) Bayesian networks inference algorithm to implement Dempster Shafer theory in reliability analysis. Reliab Eng Syst Saf 93(7):950–963

Chapter 67
Using Systems Engineering to Create a Survivable Communications System that will Operate in the Presence of "Black Sky" Hazards

Neil Siegel

Abstract Many studies have shown (and incidents like super-storm Sandy have demonstrated) that existing U.S. electronic communications systems (e.g., land-line phones, cell phones, the Internet, etc.) would not be available in the event of a large-scale, long-term power outage, whether that outage were to be triggered by terrorism, by war, or even by certain types of natural events.

However, several of the critical tasks involved in getting the power back on after such an event – such as synchronizing the restart of large-scale generators and users of electricity, allocating and planning the movement of spare parts and personnel with critical skills, and so forth – cannot be accomplished without some level of real time, wide-area electronic communications.

Because of this dependency on electronic communications and the fact that all such electronic communications systems would fail just a few hours into such an outage, in a scenario created by a group of industry experts, they found that – in the absence of some survivable wide-area emergency communications system – the duration of an outage would be measured in months or years. Casualties would be very high. When the same experts modeled this scenario while positing the existence of a survivable, wide-area emergency communications system, the duration of the outage was measured in weeks, and casualties were orders of magnitude less. It appears that the existence of such a survivable, wide-area emergency communications system is an important aspect of emergency preparations.

This paper uses systems-engineering methods to examine the question of how best to provide reliable electronic communications under such conditions at the required scale, data rates, and reliability.

Keywords Emergency response • Electric outages • Recovery from large-scale outages • Recovery from large-scale disasters • Networked low-frequency radios • Storable emergency power sources

N. Siegel, PhD (✉)
University of Southern California, Los Angeles, CA, USA
e-mail: siegel.neil@gmail.com

A.M. Madni et al. (eds.), *Disciplinary Convergence in Systems Engineering Research*, https://doi.org/10.1007/978-3-319-62217-0_67

67.1 Statement of the Problem

Many studies have shown (and incidents like super-storm Sandy have demonstrated) that existing U.S. electronic communications systems (e.g., land-line phones, cell phones, the Internet, communications satellites, etc.) would not be available in the event of a large-scale, long-term power outage, whether that outage were to be triggered by terrorism, by war, or even by certain types of natural events [1].

However, several of the critical tasks involved in getting the power back on after such an event – such as synchronizing the restart of large-scale generators and users of electricity, allocating and planning the movement of spare parts and personnel with critical skills, and so forth – cannot be accomplished without some level of real time, wide-area electronic communications.

Because of this dependency on electronic communications and the fact that all such electronic communications systems would fail just a few hours into such an outage, in a scenario created by a group of industry experts [in which the power outage was triggered by an electromagnetic-pulse attack event], they found that – in the absence of some survivable wide-area emergency communications (ECOM) system – the duration of an outage could be measured in months or years – and restoration could come in increments, rather than all at once. Casualties would be very high, comparable to large-scale nuclear war (these casualties would result primarily from sewage-borne disease). When the same experts modeled this scenario while positing the existence of a survivable, wide-area emergency communications system, the duration of the outage was measured in weeks, and casualties were orders of magnitude less. It appears that the existence of such a survivable, wide-area emergency communications system is an important aspect of emergency preparations.

The author has been leading a team that is studying this problem [2]. Drawing upon the emerging results of this study, this paper uses systems-engineering methods to examine the question of how best to provide reliable electronic communications under such conditions at the required scale, data rates, and reliability [3].

To state the problem another way: If a power outage is "big" enough (for example, all of western Europe, or the entire eastern half of the United States – and now, there are ways to cause such a large-scale outage), there is at present essentially no way to recover from this outage before casualties become very high. No organization or country has ever practiced "turning off and re-starting" a large portion of their power grid, so there is no actual experience to serve as a credible guide to the recovery effort from a power outage of this scale.

The apparent root cause for the long endurance and difficulty in recovery from such a large-scale power outage is that if a power outage is "big" enough, because of the number of generation and transmission / distribution sites involved, crews cannot get to all sites quickly, and therefore the outage will last more than a few hours. But after some period of time – certainly less than 24 h– the back-up power systems (mostly, on-site batteries) for our electronic communications systems (e.g., the internet, the land-line phone system, the cell-phone system, the ground segments of satellite communications systems, and so forth) run out of power, and those systems then shut-down [4]. Experience from periodic adverse weather events seem to indicate that (perhaps due to poor maintenance of these batteries), these back-up power systems in fact run out of power in less than 4 h.

Once these batteries run down, the electronic communications systems they support stop operating (actual experience from weather-induced emergencies actually indicates that some of these systems fail *before* their battery back-up systems run out of power, due to their being unable to cope with the large spike of usage that occurs as people try to "phone home" after the onset of the emergency). As noted above, there are many key steps in restoring electric service after a large-scale outage that are fundamentally different than restoring power after a local outage; as indicated below, these steps require real-time communications across distance and therefore need access to some sort of electronic communications system. This creates a "threshold" effect; an outage that is large enough to create conditions that make recovery fundamentally different than the city-wide or state-wide/region-wide outages that we have occasionally experienced, and the recovery mechanisms that are in place to recover from those city/state/regional types of outages will not suffice to recover from these larger outages.

To understand this aspect in more detail, recall that in general, we do not store electricity; it is used immediately upon generation. This requires that generation capacity and offered electric load be *kept in synchronicity*. Of course, electricity-consuming devices are constantly being turned on and off, but their effect on the overall power consumption within a segment of the grid is fairly small (the total use of electricity in the United States is around 5 quadrillion – 5,000,000,000,000,000 – watts per year [5]), and because the fluctuations caused by individual actions are relatively small, during steady-state power-grid operations electric use can be monitored passively at the generators, by measuring fluctuations in voltage or frequency. Generation is then matched to the real-time demand by adjusting the speed (and hence output) of generators so as to keep voltage and frequency within very small bounds of their target values [6].

Such passive monitoring, however, will *not* work for restart operations over a large area; instead, the generation station must *coordinate* in real time with the large-scale user of electricity (e.g., a water-pumping station) that is ready to come back online, so that the amount of electricity generation about to be restarted *exactly* matches the power consumption of the system about to be brought back online. Failure to do this correctly will result in significant fluctuations in voltage and/or frequency, which can damage equipment. Then, we are worse off than before; not only are things off but they are damaged. Once the small number of very large-scale users of electricity (such as water pumping, sewage treatment, natural gas pumping – which together, account for about 20% of all electricity use in the United States) are back online, bringing on small groups of residential and business users will again individually result in small (and hence, manageable) fluctuations, which can be dealt with by the normal passive-monitoring method.

But one needs some significant steady-state, precoordinated load to create a “denominator” of predictable electric load before one can start to introduce the random electric load of residences and businesses; without a big “denominator” (what matters is the *percentage* of fluctuation), the fluctuations will be too large and too fast for the passive method to control.

So, restarting the power grid after a large-scale failure requires real-time communications, but all of the electronic communications systems will be down. Analysis appears to indicate that it is not economically feasible to increase the battery back-up capacity at every point along these regular communications systems; analysis shows that we need to provide 30–60 days (not a few hours!) of back-up power to deal with the sort of large-scale power outage considered herein. Doing that at all of the millions of locations where there is electrically powered equipment for the Internet, the phone system, the cell-phone system, etc. would cost between 20 and 50 times what it would cost to provide a separate emergency communication system, which we estimate would require only about 100,000 points of presence across the United States (in contrast to the several hundred million points of presence of the regular electronic communications systems).

So it appears that we need some sort of emergency communications system that can operator on its own stored power for 30–60 days. In the rest of this paper, I use systems engineering methods to determine the necessary communications links and capacity, the necessary participants, the optimal configurations, and the appropriate restart sequencing/procedures for such an emergency communications system.

67.2 Constraints on Potential Solutions, and the Resulting Systems-Engineering Trade-Space for Candidate Solutions

Not included in this paper, but included in a longer report [7], the study team has developed what we call a “social architecture” for such an emergency communications system. This social architecture allows us to identify the users and customers for such an emergency communications system, determine how they define value within their operational context, and in general, to capture the information necessary to create a system that is both (in the terminology used in the U.S. Federal Acquisition Regulations, or FAR [8]) “effective and suitable” for their mission.

The following are a small subset of the goals and considerations from that social architecture:

- A trained operator should be able to setup the emergency communications suite at a location (e.g., to effect the transition from the long-term storage configuration into the operating configuration) in 4 h or less. This limitation of setup time will drive decisions about how the emergency communications equipment is configured for long-term storage.

- The emergency communications system should be designed so that – unless there has been physical damage sustained at a particular node – 99.9% of the locations should be able to operate (in at least a partial capacity) after being brought out of long-term storage. Achieving this level of reliability and availability will require some on-site spare parts at every site.
- The emergency communications system should be designed so that if at least 90% of the nodes in a region are brought into operation, and also at least 75% of the nodes in adjacent regions are brought into operation, then at least 75% of the emergency communications system nodes within that region should *automatically* discover a route to their regional reliability coordinator (e.g., the local emergency communications system hub). Additional nodes can achieve connectivity to the local emergency communications system hub through manual operator actions. Achieving this high level of reliability and connectivity will require the use of multiple data paths (the technical term is "communications path diversity") for most of the point-to-point linkages needs by the emergency personnel. This in turn implies a need for multiple communications devices of different types at most emergency communications system sites.
- "Connectivity" in the paragraphs above means "achieve at least push-to-talk voice and a modest data" capability.
- Push-to-talk voice communication serves a high portion of the emergency communications system use cases. The next most important capability is the ability to send a digital photograph from one location to the hub. More generalized data service is a still lower priority. These priorities will be used to establish dynamic priority-of-service within the emergency communications system.
- When we reach full deployment, there will be about 100 thousand emergency communications system installations across the United States (a full European deployment would be somewhat larger), so cost per site becomes a design consideration. An initial look suggests that the 30-day to 60-day power requirement could become the driving per-site cost element, so as the technical design evolves, taking design actions that reduce the power requirement, and creating viable strategies for lowering per-site power unit cost (but without decreasing operational availability!) will be an area of concentration.
- Although extensive preemergency planning (down to the level of checklists for individuals) will be required, no plan will survive the first hour of an actual event without requiring modification, perhaps extensive modification. Collecting status from the field is a key emergency communications system role, as this is the information required for the regional reliability coordinator (and other responsible parties at higher echelons) to adapt the plan to the actual situation on the ground and to communicate the altered and adapted plan to those in the field who will actually execute it.
- Even the relatively small number of "tier-1" restoration priorities will involve a complex web of sites, equipment, relationships, and supply chains. For example, it is probably not sufficient to get power to the stations that pump natural gas in the big main pipelines that service electric generation stations; in addition, we

will probably need to include the capability to provide power (perhaps by emergency batteries or generators) to whatever control station remotely plans and operates the valves and pumps on those big natural gas pipelines. Each item included in tier-1 (and the other tiers, too, of course) will have a similar web of such relations and a tiered supply chain that need to be thought through: people, power, spare parts, procedures, and so forth.

- At present, neither the United States nor any European country has such an emergency communications system; therefore, building consensus that one is required is an essential step. We believe that voluntary actions are vital to getting the process started of building such consensus underway, but at some point, we believe that we will need federal statutes and/or regulations to enforce compliance and help convince utility-oversight boards that it is legitimate to recover emergency communications system costs through utility rates, and so forth. We point out that seat belts and car pollution control were not widely adopted until government starting creating mandates (in those cases, first the state of California and then the federal government).The way the cyber protection story is playing out reinforces our belief that mandates will eventually be required in that field, too, but that voluntary efforts will be needed in order to create the environment necessary to get those mandates enacted and to figure out what the correct form and content of those mandates ought to be. We believe that the process of building a societal consensus to build such an emergency communications system will have to go through a similar process.

We next mapped the goals, requirements, and considerations developed in the emergency communications system social architecture into candidate solutions. This process was undertaken herein through several steps:

- Identify candidate communications technologies
- Create a list of key issues/risk areas
- Using that list of key issues/risk areas, identify a set of key technical trade studies
- Create a set of candidate designs, together with methods and metrics for selecting among those candidates designs. Make the initial design selection, provide the rationale, and make a preliminary assessment of the feasibility and performance of the selected design.

These are discussed in the following sections.

67.2.1 Identify Candidate Communications Technologies

Based on our experience implementing many sorts of high-reliability communications systems [9], we selected the following technical methods of communications as the candidates for consideration within this emergency communications system:

- HF radio, supplemented with some sort of networking
- VHF radio, supplemented with some sort of networking

- Higher frequency radio, supplemented with some sort of networking
- Meteor-burst radios, supplemented with some sort of networking
- Use of power lines to carry communications signals
- So-called dark fiber, that is, fiber optic communications cables that are installed but not in active service
- Commercial satellites (low-Earth orbit)
- Commercial satellites (geo-stationary orbit)
- Various combinations of the above

67.2.2 Create a List of Key Issues/Risk Areas

Given the results of the social architecture and the above list of candidate technologies, we identified the following as key risks that need to be addressed through the technical trade study process for the emergency communications system:

- How to provide power for the emergency communications system at each location for the specified 30-day to 60-day period?
- What spectrum (RF frequencies) would be available for the emergency communications system during emergency operations? Not all of this spectrum allocation need be available for use during nominal (nonemergency) use.
- What techniques and materials would allow the emergency communications system equipment to be stored for long periods of time (years or decades), yet still be periodically tested, maintained, and support periodic training?
- How to allow the emergency communications system to adapt, ideally almost automatically, to the likely differences between the anticipated emergency conditions and those that actually come to pass?

Many other risks were identified, but there were the ones that through our assessment process represented a combination of likelihood and potential impact that stood out as potential key system disablers if not properly addressed and mitigated.

67.2.3 Using that List of Key Issues/Risk Areas, Identify a Set of Key Technical Trade Studies

Given the above list of key risk areas, we then identified the following as the key trade studies that we needed to perform:

- How to provide 30-day self-contained power, with a long storage life, reasonable maintenance requirements, and high availability at need?
- Spectrum availability during emergencies, and based on that, selection of the actual communications mechanisms, and eventually, the actual devices

- Approaches and materials to achieve effective long-term storage of the emergency communications system equipment
- Techniques to support self-adaptation of the emergency communications system

The principle results of these trade studies led to a candidate solution, presented below. Reasonable technical approaches were found to exist to mitigate all risk areas. We were not provided with an explicit “design-to” per-site cost, but we believe that most of the study outcomes will allow options to be selected that are deemed reasonable in cost. One area that still needs cost-per-site optimization is the 30-day to 60-day self-power requirement; we have excellent options, but the specific mix of technologies will probably vary significantly with the specifics of the site (e.g., is the use of solar panels as augmentation at this site feasible and attractive?), and therefore it is too early to roll up this portion of the cost to a system-wide total; actual regional site surveys will probably be required in order to arrive at credible system cost estimates.

67.3 Candidate Solution, Arising from the Systems Engineering Studies

In this section, we describe the candidate design and the initial work that has been undertaken in order to validate the efficacy of that design.

In our work to-date, we have gone as far as selecting many key technical parameters (such as radio frequencies and transmission polarizations), but not all technical parameters (e.g., we have not yet selected exact RF power levels, antenna sizes, and manufacturers; that latter level of detail will come later). But we have progressed far enough to demonstrate the technical feasibility of the proposed design, and to estimate some of the key system parameters, such as availability, network connectivity rates, and storage life.

The following are the components selected for use within the emergency communications system:

- The following devices are at each emergency communications system location:
  - HF Near vertical incidence skywave (NVIS) radios, with small magnetic antennas
  - UHF radios
  - Packet routers
  - Software to implement and control the above functions
  - A power subsystem, based on vanadium redox flow batteries, with control circuitry to enhance reliability, supplemented at many locations by solar panels, and perhaps supplemented at a small number of sites by wind-powered generators.

- The following device is located at each regional reliability center:
  - A packet router equipped with special, mission-specific software agents
- The emergency communications system can optionally include the following components:
  - Mobile emergency communications system nodes (e.g., trucks or other vehicles that are equipped with emergency communications equipment, a packet router, and appropriate power equipment).
  - Mobile emergency power systems. These are trucks that have battery-power subsystems that can be driven to power an emergency communications system location whose batteries have failed. They can also be used, of course, to power other types of equipment in case of need.

67.3.1 The Emergency Communications System Design Story

- These Site-to-site communications are provided by the HF radios and the UHF radios. The disadvantage of large size traditionally associated with HF radio antennas is corrected through the use of magnetic antennas. UHF provides very high-quality service, but at a shorter range than HF, while HF radios can operate beyond line-of-sight. Hence, we include *both* frequencies in the emergency communications system design.
- Thirty to sixty days of stand-alone power is provided at each site. In order to be affordable, the size of the stand-alone power array will vary from site to site, driven by a cautious estimate of emergency communications system power requirements for 30–60 days. At many sites, the size (and cost) of the battery array is decreased by the addition of solar panels. There is the potential to add wind power at a few, suitably situated, sites.
- Vehicle-mounted emergency communications system configurations are possible. The UHF component could be configured to operate on-the-move; the HF NVIS component will likely be configured only to operate at-the-pause (e.g., when the vehicle is parked); this limitation is due to the need for the UHF antenna to be 15–20 ft. above ground level, implying that the extendable mast must be erected for HF NVIS coverage to work. Power for the emergency communications system equipment on these vehicles will be provided by a combination of batteries and enhanced vehicle alternators.
- Truck-mounted emergency communications system portable battery configurations are possible, providing a portable power source that can be moved from site to site during an emergency. This could power emergency communications system equipment at a site where the battery has been damaged, for example, but could also be used at any site to provide temporary emergency power.

- The single-hop, direct site-to-site communications success rate is improved through the use of error correction coding and other higher level communications protocols. These are implemented in the packet router located at each emergency communications system site.
- At each site, there are two independent radios, on different frequency bands, utilizing different modulations. This provides a basic type of *communications path diversity* and thereby improves system reliability. The router at each site determines which radio to use for each transmission attempt (whether voice or data), based on its radio "visibility" to adjoining sites. No manual action is required by the emergency communications system user to select the best radio for each transmission; this is accomplished for them automatically; we do not expect the emergency personnel to be radio frequency propagation experts!
- The packet router also uses the same visibility information to implement multi-hop communications for both voice and data: a communications link need not be "direct"; I can talk to you through a set of intermediate nodes, e.g., the data are in fact routed through other emergency communications system sites. The finding and utilization of such paths are automatically accomplished by the packet routers; no manual action is required by the emergency personnel to find and implement such multi-hop paths.
- Frequency selection is based on time-of-day, atmospheric conditions, and other factors. An "intelligent director" (a packet router equipped with mission-specific software agents) controls and coordinates this process, providing direction to the packet routers, which in turn command the radios to use the appropriate frequencies and other radio settings. No manual action is required by the emergency personnel to account for day/night frequency preferences.
- The over-all policy for frequency utilization must be coordinated with regional and national civil officials. This also is implemented in the intelligent director; no manual action is required by the emergency personnel user to comply with the ever-changing radio frequency policy.

Figure 67.1 depicts the methodology used to provide a preliminary validation of some of the key technologies selected for emergency communications system (ECOM). On the left side of Fig. 67.1, you can see that we draw upon the requirements and goals for the emergency communications system (as developed and described in the social architecture section, above) as the criteria against which we measure candidate designs. You can also see that we use the specified performance for the products selected as part of the inputs to our emergency communications system performance model.

A system performance model makes various assumptions about certain aspects of the system it is modeling. By drawing upon actual performance achieved by similar systems, we can create a model that is of high credibility.

The author was the program manager for the U.S. Army "Blue-Force Tracker" [10]. This system uses VHF and UHF radios that are interconnected via local routers to implement its communications network, and therefore, is in some critical ways similar to the communications systems proposed herein for ECOM; since the Army system was fielded in 1999 [11] and has been in successful and continuous

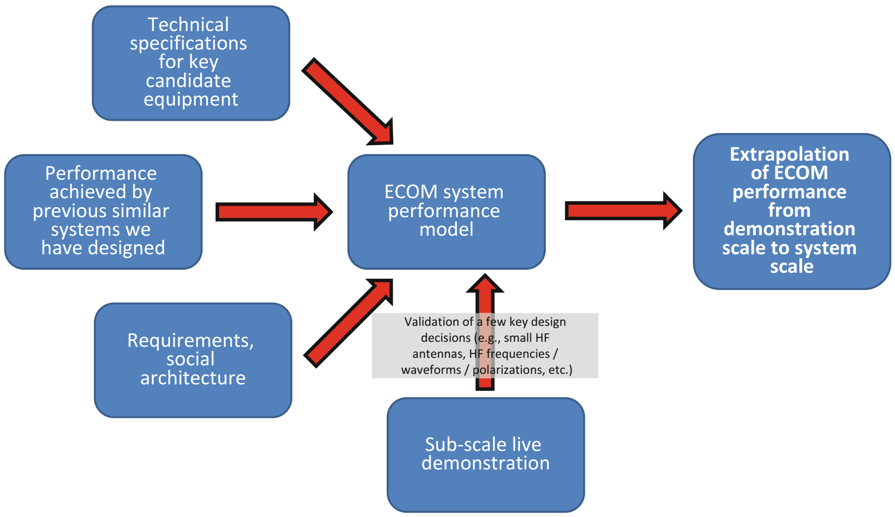

Technical specifications for key candidate equipment
Performance achieved by previous similar systems we have designed
Requirements, social architecture
ECOM system performance model
Extrapolation of ECOM performance from demonstration scale to system scale
Validation of a few key design decisions (e.g., small HF antennas, HF frequencies / waveforms / polarizations, etc.)
Sub-scale live demonstration

Fig. 67.1 Process to achieve preliminary validation of the selected design for the emergency communications system

operational use since that time [12], this means that many of the key technical and architectural features proposed for ECOM have been validated through their use on that system.

The motivation for the design of this Army system was similar to the goals for this emergency communications system: very high reliability was required, and the design solution was to achieve this via *communications path diversity*: multiple radios, on different frequencies, using different waveforms, with the best path determined in real time by the attached routers [13]. In one cardinal way, the Army system was much more difficult than this emergency communications system: most of its nodes have to operate while *on the move*, and therefore, had to deal with continuously changing line-of-sight interruptions, interruptions caused by terrain masking and foliage masking, and so forth. Most of the operating locations for this emergency communications system are, in contrast, at fixed sites, and therefore this system does not face these difficulties. On the other hand, the sites for this emergency communications system are on average farther apart than the units for the Army system; this is what led to the selection of HF radios for use within ECOM, as HF can achieve longer single-hop communications distances than the VHF and UHF radios used in the Army system.

At the bottom of Fig. 67.1, you see that we actually conducted a subscale live demonstration of certain critical aspects of our candidate design; specifically, the HF radios and their magnetic antennas. We selected these components for a subscale live demonstration because they are the principle component-level difference from the Army system mentioned above. This demonstration validated that our assumptions about the performance of HF radio with the magnetic antennas were correct. We also did a live performance measurement field test with a pair of UHF radios.

Quantitative data about HF NVIS radio performance were also collected. Given the ranges and packet-completion rates provided therein for HF NVIS, the measurement results with the UHF radios [together with the knowledge we have from the Army system about comparable radio performance data for UHF], we made estimates for the performance of ECOM at the system level using system-modeling methodologies validated through our work on the Army system (whose architecture is similar). These results show that the candidate emergency communications system design will perform very well, indeed.

67.4 Summary, Portions of the Problem Remaining Unsolved, Candidate Next Steps

The combination of social architecture, operational mission threads, and other analyses have led to insight regarding both what will make the subject emergency communications system both "effective" and "suitable," using the terms from the Federal Acquisition Regulations. The work described herein to capture the data needed by the emergency personnel will allow the development of an effective system, whereas the social architecture (and the implications derived from it) will allow the development of a suitable system. As noted in the discussion, we believe that the technical challenge of creating a *suitable* system is in the case of this emergency communications system the *harder* part of the problem, and therefore, we have placed more emphasis on that portion. A set of specific design features – role-based processing, remote authentication, automatic management and configuration of the radio network, and so forth – have been identified as high leverage for the emergency communications system user community: both those *coordinating* the recovery operation and those actually *implementing* the recovery operation.

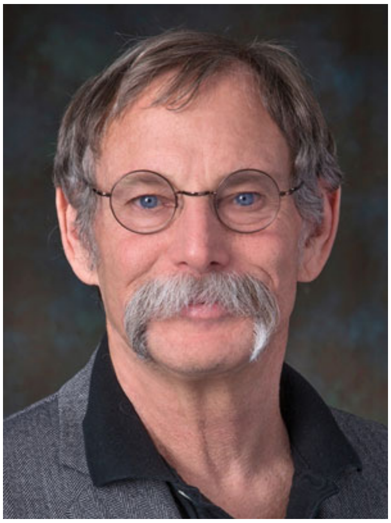
Of course, the emergency communications system design, development, and deployment effort is just beginning. Consensus (and funding) must be acquired across a wide range of stakeholder communities. Prototyping – in order to create tangible artifacts showing what the emergency communications system would look like, and how it would operate – is probably essential in order to help build this consensus. The following provides a list of candidate next steps for the emergency communications system:

- Implement a pilot (e.g., smaller scale) implementation of the emergency communications system. This would create a tangible artifact that could support demonstrations. Experience suggests that this is very important in terms of building support for the implementation of the system, in addition to the technical value that would result from the learning that would come from building the pilot system. Use the technical lessons learned from the pilot to finalize an actual scalable detailed design
- Build and exercise the emergency communications system system-level performance model, described in Sect. 67.3.

- Continue work on the concepts for implementing the organizational relationships and coordination mechanisms required for using the emergency communications system
- Continue work for the manning plan: for example, training regular utility company employees and also creating a surge capacity concept that can bring in needed personnel from other industries
- Build and utilize an emergency communications system system-level cost model that allows for estimates about number of sites, hours of utilization per day, solar availability, and other factors to be adjusted, so as to create parametric estimate curves for emergency communications system cost. Estimate the cost and time of implementation. Suggest phasing and incremental capabilities
- Investigate concepts for funding (acquisition, training, and maintenance) and cost recovery, for example, who would pay, how might cost be shared, how could costs borne by the operators and owners of critical infrastructure recover those costs through utility rates, and so forth
- Develop ideas for artifacts that would help other stakeholders understand the problem and the proposed solution (so as to help build a constituency that would support the implementation of such a national emergency communications system)
- Consider the creation of some use cases that would use some of the emergency communications system equipment in nonemergency or lesser emergency situations, in order to help reinforce the business case

Acknowledgments This work was funded in part by the Electric Infrastructure Security Council, a nonprofit research entity established by the electric utilities and their major customers, whose support is appreciated.

About the Author



Neil Siegel is the IBM Professor of Engineering Management in the Department of Industrial and Systems Engineering at USC. Previously, he was Sector Vice-President and Chief Technology Officer at Northrop Grumman for 15 years, and before that, he held a variety of senior leadership positions within that company. He holds more than 20 patents, and his inventions are used in a billion devices worldwide. He has been elected to the U.S. National Academy of Engineering and received the Simon Ramo medal for systems engineering and systems science. He is a fellow of the IEEE and an INCOSE-certified Expert Systems Engineering Practitioner and has received a variety of other awards and honors.

References

1. Note that the U.S National Academy of Science and the U.S. agency NASA now estimate that the probability of an adverse space weather event (e.g., solar-induced electronic storms severe enough to cause continent-scale electric power outages) in the next decade is around 12%. See http://sites.nationalacademies.org/cs/groups/ssbsite/documents/webpage/ssb_153147.pdf and https://science.nasa.gov/science-news/science-at-nasa/2014/23jul_superstorm
2. Some of the preliminary results of this study are available at http://www.eiscouncil.com/Library; see the item “BSX White Paper” on that web page
3. This paper is one of four that is part of a coordinated CSER conference session that is considering the topic of “Protecting Electric Power Sources”. This panel was chaired by the author of this paper
4. Generally, the back-up power devices for these systems are designed to provide between 4 and 8 hours of operation
5. For example, see information from the U.S. Energy Information Administration (http://www.eia.gov/electricity/annual/)
6. In the U.S., the target values are the familiar 120 volts of alternating current, alternating at a rate of 60 cycles per second. In Europe, 220 volts and 50 cycles per second
7. “Emergency Communications System (ECOM), A Technical Report for the Electric Infrastructure Security Council”, by Neil Siegel, Ph.D. and Bran Ferren, 8 August 2016
8. Available on-line at https://www.acquisition.gov/?q=browsefar
9. Including the emergency communications system in the City of New York, and the U.S. Army’s “tactical internet” and Blue-Force Tracker. See “The Digital Battlefield: A Behind-the-Scenes Look from a Systems Perspective”, Neil G. Siegel and Azad M. Madni, Elsevier, 2014
10. Formally known as “Force XXI Battle Command Brigade-and-Below”, or FBCB2
11. “Force Tracking – Experience in Combat”, Neil Siegel, 2005
12. “The Digital Battlefield: A Behind-the-Scenes Look from a Systems Perspective”, Neil Siegel and Azad Madni, Elsevier, 2014
13. “Digitizing the Battlefield”, Neil Siegel, a chapter in the book *Fateful Lighting*, information technology association of America, 2002

Chapter 68
Interdependency Effects on the Electricity Grid Following a "Black Sky" Hazard

Jonathon E. Monken

Abstract The protection of the bulk electric system in the face of large-scale threats like electromagnetic pulse and geomagnetic disturbances poses several unique systems engineering obstacles. This class of hazards, known as 'Black Sky' hazards, trigger unprecedented outages due to the scale of the triggering phenomena. Traditional risk assessment models and recovery plans have core assumptions that generation, transmission and distribution are unlikely to be simultaneously impacted and that physical damage will be relatively localized; based on these assumptions, industry can access power from neighbouring grid networks and restore some level of service fairly quickly, even if the localized problem takes far longer to correct. In contrast, outages at the scale caused by these 'Black Sky' hazards cannot be addressed by these traditional techniques. Before we develop effective recovery strategies from such 'Black Sky' events, new insights about root causes at such a large scale are required. Some of these insights are provided in this paper.

We start with an overview of the operations of one Black Sky triggering event, an electromagnetic pulse attack showing how such an event could operate at a scale of impact that is far larger than anything experienced.

We then show that, due to the interdependencies between the electricity, natural gas and communications infrastructures, the large scale of the EMP triggering event introduces failure modes not experienced in previous power outages. We analyse some of the key failure modes that will occur in outages of this scale and show that systems and processes developed for recovering from more common hazards will not be able to correct these new types of failures. We will also show that attempting to recover using these conventional procedures actually has the potential to damage additional equipment, placing recovery farther away than ever.

We provide a more nuanced understanding of the natural gas delivery system that we believe will be vital to curbing the potential effects of major pipeline disruptions resulting from a hazard that precipitates an outage combined with the second-order effects from the loss of electricity to refining and pumping operations.

J.E. Monken (✉)
PJM Interconnection, LLC, Norristown, PA, USA
e-mail: Jonathon.monken@pjm.com

A.M. Madni et al. (eds.), *Disciplinary Convergence in Systems Engineering Research*, https://doi.org/10.1007/978-3-319-62217-0_68

Keywords Emergency response • Electro-magnetic pulse • Large-scale disaster • Interdependency

68.1 Statement of the Problem

The protection of the bulk electric system (BES) in the face of large-scale threats that directly affects electronics like electromagnetic pulse (EMP) and geomagnetic disturbances (GMD) poses several unique systems engineering obstacles. This class of hazards, known as 'Black Sky' hazards, create unprecedented outages due to the scale of the triggering phenomena. Traditional risk assessment models and recovery plans rely on core assumptions that generation, transmission and distribution are unlikely to be simultaneously impacted and that physical damage will be relatively localized. Based on these assumptions, industry could follow standard practice and access power from neighbouring grid networks to restore some level of service fairly quickly, even if the localized problem takes far longer to correct. This methodology allows rapid (less than 72 h) recovery of a large percentage (more than 75%) of outages and minimizes the number of major infrastructure systems affected. This was the case even in a large-scale event such as Superstorm Sandy, where 70% of the 8,511,251 peak customer outages were restored within 5 days of the storm [1] making landfall was met in large part due to access to electricity outside the affected area and a minimal impact to bulk generation assets. Common practice for grid restoration follows a market and economic-driven approach of providing electricity to the maximum number of customers in the minimum amount of time. Response activities typically include a concerted effort to restore connectivity to targeted critical infrastructure; however, the criteria for what classifies as critical are generally subjective and changes based on local government priorities. This methodology requires access to adequate generation assets (and their fuel sources) and power transmission to meet all load requirements, in addition to having sufficient numbers of trained personnel and equipment. In short, the means of restoration is largely dependent on the breadth of impact and the availability of the equipment and assets classified as 'long-lead' due to their relative rarity and complexity of installation, high-voltage transformers, large generation assets and the transmission lines providing the connective tissue. Under these more 'routine' circumstances, the unprecedented challenges that accompany an outage with widely distributed damage lasting weeks or months are mostly avoided, making them poorly understood and inadequately planned for.

By contrast, outages at the scale caused by 'Black Sky' hazards cannot be addressed by traditional techniques due to the inherent complexity of their rarely experienced impacts. The defining characteristics of Black Sky hazards are a large geographic footprint and duration measured in weeks and months instead of hours and days. While the breadth of impact is largely attributable to the nature of the precipitating event, the duration is a combination of the scale of the hazard and a cascading impact on interdependent infrastructure systems essential for power generation and grid operation. This manifests in the form of (1) load shortage—typically the result of extensive damage to the distribution system thus reducing demand to balance generation, (2) an inability to balance load—likely due to insufficient connectivity within the transmission system or (3) inadequate capacity of electricity to deliver because of damage to generation assets or a lack of fuel to operate. Any one of these factors experienced over a large enough area or for a long enough period of time necessitates a fundamental shift in how grid restoration is conducted. Instead of the standard customer and market-driven methodology described earlier, the electricity industry is forced to focus on protecting the backbone of the bulk electric system (BES) itself in order to restart the system from within, a process known as ‘black start’. Designated black start generators are intended to serve as the ‘islands’ of electricity upon which the rest of the grid will be restored beginning with the transmission corridors that connect them, known as ‘cranking paths’. Determining the amount of generation included in the black start plan is usually the responsibility of the independent system operator (ISO) or the regional transmission operator (RTO) (Fig. 68.1), who operates as a balancing authority within the three larger interconnects in the United States (Fig. 68.2).

The electricity demand served by the black start plan, known as ‘critical load’, consists of three primary categories of load customers: (1) nuclear power stations in order to ensure their ability to go into safe shut down during an emergency, (2) ‘Hot start’ generation plants that would be damaged without near-term restoration and (3) natural gas infrastructure, a term which is broadly defined and varies regionally. This process does not allow much emphasis on power needs outside of the electricity sector and creates a circumstance where other life-support infrastructure systems are highly unlikely to be restored in a timeframe that aligns with any of their contingencies for alternate, back-up power. The result is a prisoner’s dilemma,

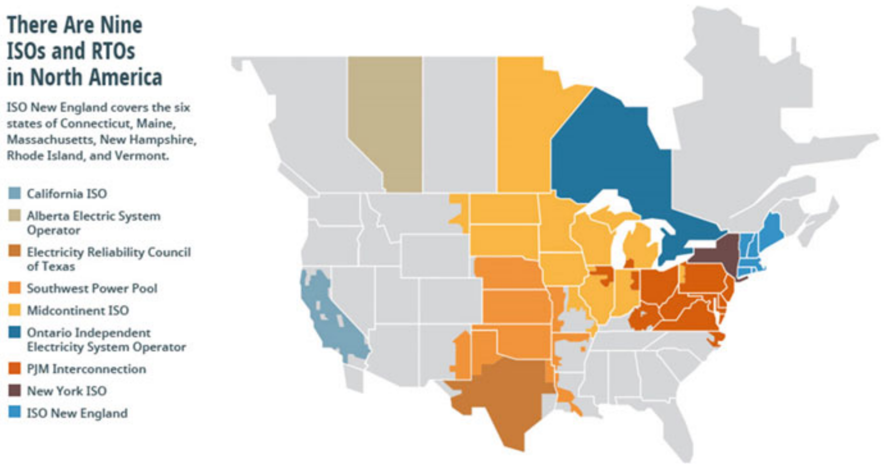

There Are Nine ISOs and RTOs in North America
ISO New England covers the six states of Connecticut, Maine, Massachusetts, New Hampshire, Rhode Island, and Vermont.
California ISO
Alberta Electric System Operator
Electricity Reliability Council of Texas
Southwest Power Pool
Midcontinent ISO
Ontario Independent Electricity System Operator
PJM Interconnection
New York ISO
ISO New England

Fig. 68.1 Map of the nine independent system operators (ISO) of North America

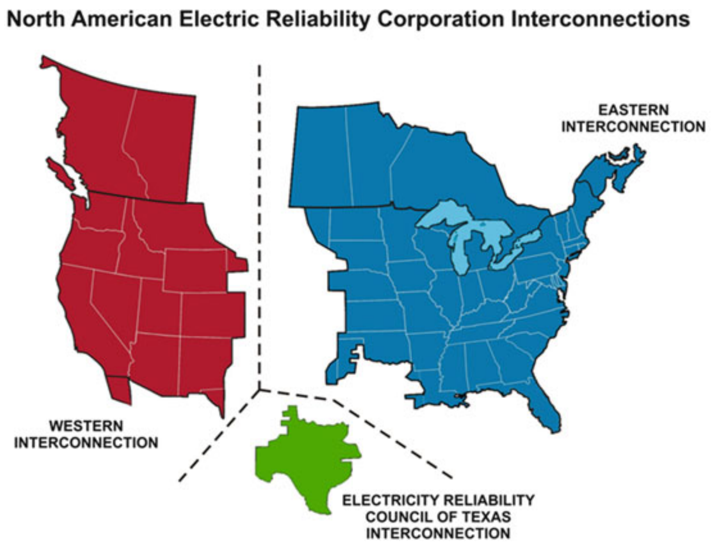

North American Electric Reliability Corporation Interconnections
EASTERN INTERCONNECTION
WESTERN INTERCONNECTION
ELECTRICITY RELIABILITY COUNCIL OF TEXAS INTERCONNECTION

Fig. 68.2 Map of NERC interconnections

where the decisions made by the electricity industry intended to preserve their internal requirements are done potentially at the cost of system they depend on. In the case of electricity delivery, the two most vital dependencies for effective operation are the communications systems needed to operate power flows over the grid and access to the primary fuel sources of generation. Any significant disruption of developing effective recovery strategies from such 'Black-Sky' events, new insights about root causes at such a large scale are required.

68.2 Constraints on Potential Solutions, and the Resulting Systems-Engineering Trade-Space for Candidate Solutions

In order to solve the problems associated with energy restoration in a Black Sky event, it is important to understand the extent of interdependencies with communications and natural gas infrastructure in both a technical and an operational context. Additionally, the context of a particular hazard, in this case an electromagnetic pulse (EMP) attack and its unique effect on electronics, will demonstrate the complexity of the these constraints. Without this working knowledge, a myopic focus on the grid would not produce results that can be implemented successfully.

68.2.1 Communications Interdependency

Reliable and efficient delivery of electricity requires the careful balancing of load over vast distances between thousands of substations utilizing sophisticated industrial control systems (ICS) and supervisory control and data acquisition (SCADA) systems housed in control centres (Fig. 68.3).

The substantial majority of all substations are operated remotely, making them completely reliant on Internet, voice or fibre connectivity in order to inform a control centre of voltage levels, the status of breakers and relays, and the overall health of components [2]. This connectivity also includes the ‘dispatch’ of generation assets, which consists of remote commands brining more or less electricity online to match demand. Without access to these automated systems, manual operation is required to provide timely readings of demand and response signals at substations, and to locally operate breakers and equipment needed to alter power flows. This form of operation is not only logistically challenging due to personnel constraints but the lack of real-time data in such a dynamic environment coupled with the comparative delay in operator activities means it is impossible to achieve the same level of grid efficiency in such circumstances and is likely to result in additional outages. While some redundancies exist within the industry to provide a means of back-up communication, there is a significant reliance on commercial communications systems that require access to sustainable grid electricity in order to function at a level needed to connect the thousands of endpoints in the grid network (Fig. 68.4).

The information carried by the communications networks is not limited to commands for grid operations, it also includes large quantities of data [3] utilized for the creation of load forecasts used to anticipate future power requirements and to operate the energy markets that sustain the financial models of the companies connected to it. While the operation of the grid is prioritized over the operation of the markets and data flows can be reduced to a quantity that is limited to critical functions, the loss of complete data will lead to inefficiencies that could threaten the entire system if the outage persists. Additionally, the grid is broken into

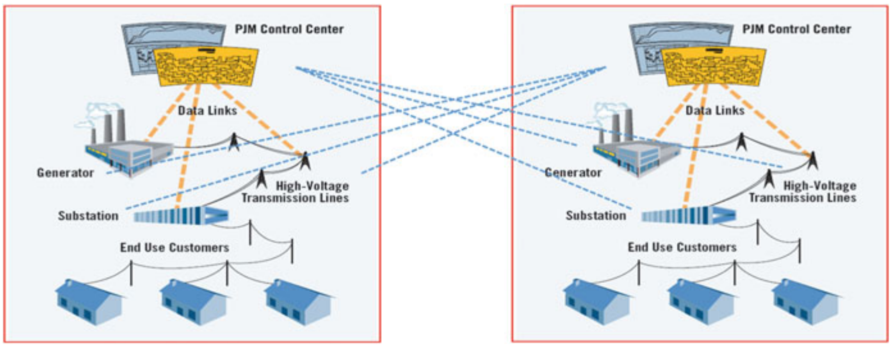


Fig. 68.3 PJM interconnection redundant control centre model for grid optimization

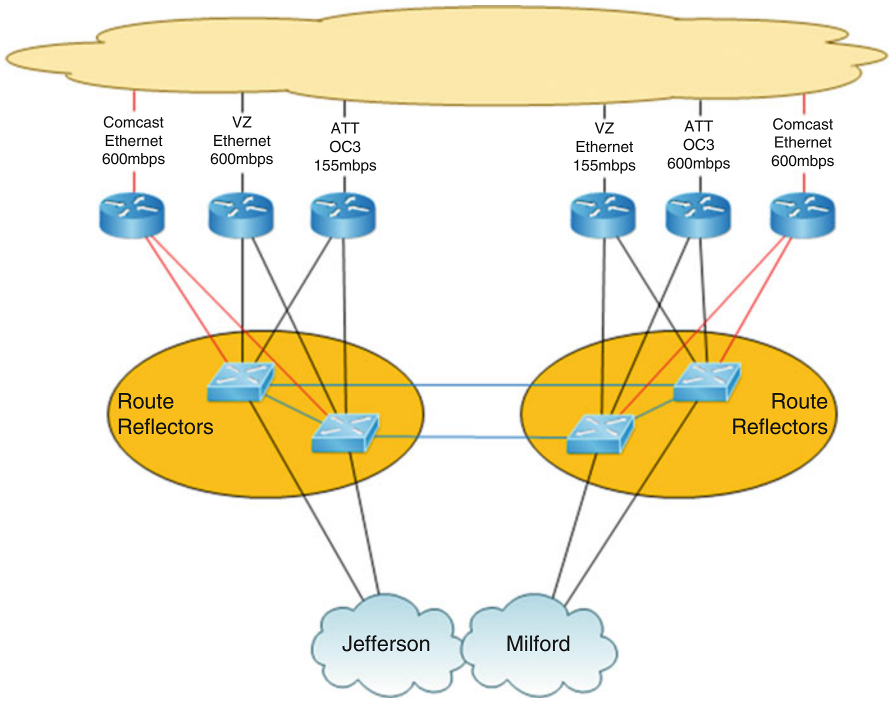


Fig. 68.4 PJM control centre Internet connectivity for grid operations

transmission operator zones, marking a segment of operational data known as state estimators (SE), which essentially aggregates the data and provides the means to balance the input and output of each zone. The loss of a single state estimator will immediately affect the system, but the extent of impact is dependent on both the relative size of the transmission operator zone and the duration of the outage. In the event that multiple SEs are down or one is down for an extended period of time, the overall balance of the system will continue to drift towards inefficiency and the unavoidable result would be outages if SEs are not restored.

The backbone of these systems is a highly interconnected network of cellular sites, switching stations and data centres used to operate and control traffic over the various lines of communications. These sites are widely distributed and completely dependent on electricity, and while control centres are well equipped to function on back-up generation for a period of 3–5 days, the communications infrastructure they rely is not as well equipped to handle long-duration outages.

68.2.1.1 Key Issues, Risk Areas and Required Information

There are several topics of research needed in order to conduct a systems-level analysis of the communications sector's interdependency with the electricity industry. Ongoing studies at organizations like the North American Transmission Forum's (NATF) Spare Tire project and the Electric Infrastructure Security (EIS) Council's BSX project [4] identify some of the broad-level issues in this space. Both projects indicate that currently there are insufficient redundant communications systems to operate in this type of environment to operate with adequate sufficiency to operate in a Black Sky environment. However, more work within the electricity sector is needed to determine:

- How many communication end points within the electricity sector need to be connected in order to execute black start and operate the grid with enough efficiency in order to service critical load already identified?
- What is the type and quantity (bandwidth) of voice and data needed in order to accomplish the aforementioned critical tasks?

This information is vital to determining the base-level functionality of a dedicated communication system to operate in this environment.

68.2.2 Natural Gas Interdependency

Second only to a loss of communications, the greatest threat to the successful operation of the electricity is the loss of its primary fuel source for generation. While there is a relatively diverse base of fuel sources currently available, there are limitations of each fuel to function in a Black Sky environment and the trend line for fuel mix is rapidly changing (Fig. 68.5).

As noted in the figure, natural gas surpassed coal as the largest source of generation in the PJM market in 2015 and is currently on pace to produce a substantially greater portion of the total within the next 3–5 years. The trend is largely the result of the dramatic increase in the availability of natural gas due to hydraulic fracturing resulting in significantly lower prices in the United States, and new emission standards for carbon output in electricity generation. This is especially relevant given the particular weakness of natural gas generation as a black start plant as compared to other fuel types when evaluated based on the criteria of fuel security, variable load functionality and overall reliability on a scale of 1 (low) to 5 (high):

Table 68.1 Definitions

- Fuel security–On-site storage and the susceptibility of supply to be interrupted
- Variable Load—Ability of generation station to raise and lower output to meet demand
- Reliability—Track record of meeting generation dispatch when called upon

Nuclear generation sites typically have up to 18 months of fuel stored on-site and operate at a high level of reliability, but they do not have any variable load capability and are not authorized to operate without access to external grid

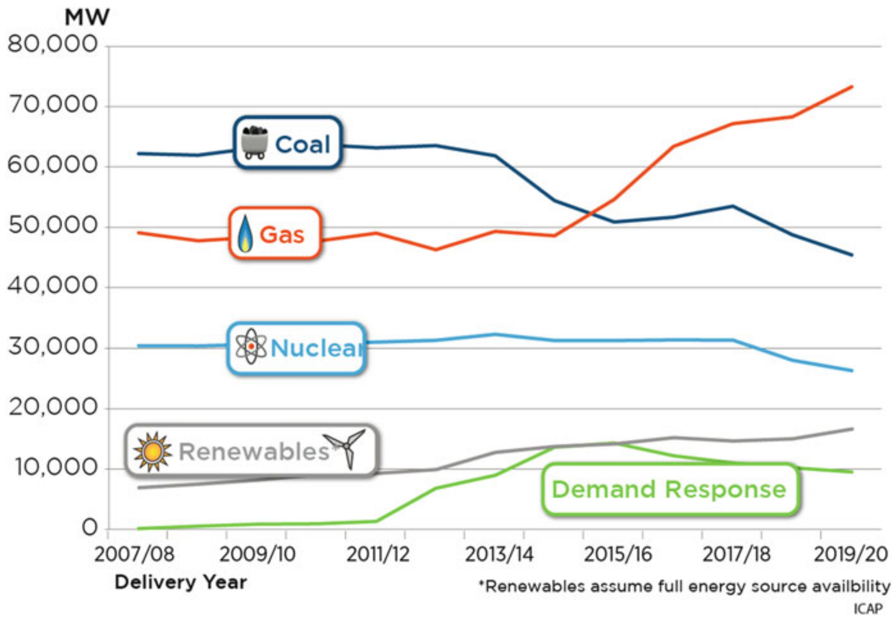


Fig. 68.5 Trend line for fuel sources in PJM market

Table 68.1 Black start capability assessment

Black start capability	Fuel security (*t*)	Variable load (*t*)	Reliability (*t*)
Nuclear	5	1	4
Coal	4	5	5
Renewable	3	2	3
Natural gas	3	5	5

electricity. Renewable sources such as wind and solar have some variable load capability under certain configurations, and when combined with storage, their fuel source is dependent on weather conditions and they are not as consistently reliable as other sources. Coal routinely maintains up to 6 months of fuel stored on-site and has an excellent record for reliability and variable load functionality, but the number of sites is dropping rapidly in response to the low cost of natural gas and the higher rate of carbon emissions. By comparison, natural gas matches the reliability and variable load capability of coal but does not have a comparable ability to maintain a reserve of fuel on-site due to the cost and environmental constraints, instead relying on just-in-time delivery of natural gas through pipelines.

Compounding the challenge is the structure of the natural gas transmission industry, the accepted norms of restoration prioritization and the sector's growing dependence on electricity to extract, refine and transport gas through each stage of delivery. The industry is far more fragmented than electricity, with no comparable physical network of interconnections or regional-level transmission operators to

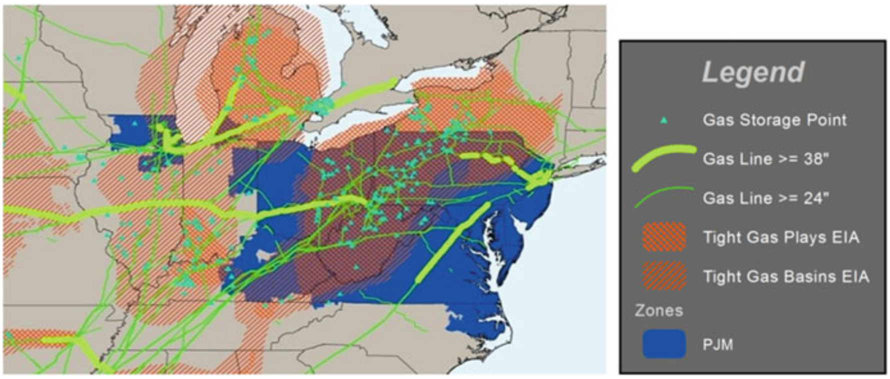


Fig. 68.6 Natural gas local distribution companies in PJM territory

ensure reliable delivery. However, the greatest similarity for operations is a reliance of ICS and SCADA in order to manage flow in the network. Much like the electric grid, the vast majority of nodes throughout the delivery system are unmanned, instead relying on a network of remotely monitored sensors and controls to provide services. The system is broken in Local Distribution Companies (LDCs), with 23 LDCs in PJM’s territory alone to coordinate with (Fig. 68.6).

Throughout these LDCs, there is network of storage and pumping stations, along with compression stations along the pipelines that keep the flow of fuel moving. All of these systems are becoming increasingly reliant on electricity in order to function. While most of the compression stations rely on in-line fuel in order to run, an increasing percentage of them are converting to electric compressors in order to decrease their carbon emissions and improve their overall efficiency. Without continuous access to grid electricity, their back-up diesel generators cannot run for extended periods of time. Compounding the challenge is the method by which gas companies prioritize their supply of natural gas in circumstances where the volume of natural gas is reduced, known as ‘tight’ conditions. First priority goes to homeowners, given the common use of natural gas for heating and cooking. Unless generation companies incur additional costs for secure ‘firm’ service contracts to be placed at the front of the line for delivery of natural gas, they fall towards the bottom of the list when production is limited. Without the just-in-time delivery of fuel, pipeline-fed natural gas generation stations cease to function within hours.

68.2.2.1 Key Issues, Risk Areas and Required Information

Given this set of constraints, there are several key areas of discovery needed to advise a data-driven systems engineering analysis of the natural gas sector as it pertains to its share of the larger challenge of Black Sky restoration. The following information is needed:

- How many communication end points within the natural gas sector need to be connected in order to execute black start and operate the pipelines with enough efficiency in order to service black start generators already identified?
- What are the minimum service levels of gas delivery needed in order to adequately supply the black start generators?
- Identify and aggregate the critical loads of natural gas infrastructure (refinement, storage pumping, compression stations for distribution) for service in an outage environment.

68.2.3 The Electromagnetic Pulse Threat

The greatest collective threat to these systems is a hazard that sits at the intersection of their shared vulnerabilities. The case study for this analysis will utilize an electromagnetic pulse (EMP) event, which is caused by the E1 pulse generated by the high-altitude detonation of a nuclear weapon. The effects of such an attack would be profound, and the infrastructures discussed in this paper would be significantly affected. The primary vulnerability to EMP is dense clusters of low-voltage electronics, something found in abundance in the control centres, data centres and substations of the three systems discussed. Additionally, the geographic footprint affected by an EMP attack is consistent with the hallmark of a Black Sky event in that it is large enough to impact a substantial portion of the continental United States simultaneously. The failure of electronics as a result of the E1 pulse varies based on a variety of factors including their protection against such an effect and their relative proximity and line-of-sight to the pulse, but the compound effect of so many components in so many systems being affected simultaneously is likely to be catastrophic. Even the process of conducting the engineering assessment of what specific components are damaged following an attack is an activity that would take far more time to complete than what is acceptable for standard restoration operations. In the case of the electricity industry, the most important task to complete in the early stage of a response is to rapidly assess the overall health of the system. This activity is not only vital to determining the extent of damage but to avoid causing undue harm to the system by improperly balancing load and further damaging critical components that were unharmed from the initial event. To compound the issue, the communications systems (and their back-ups) needed to conduct this rapid assessment are certain to be affected and will have significant operational limitations in this environment largely due to their constraints on back-up power (Siegel, 2017) [5]. Even if the electrical grid achieved an acceptable level of restoration, the control systems employed by the natural gas transmission industry are certain to be impacted as well and cannot ensure uninterrupted fuel delivery to the generation sites needed to execute black start.

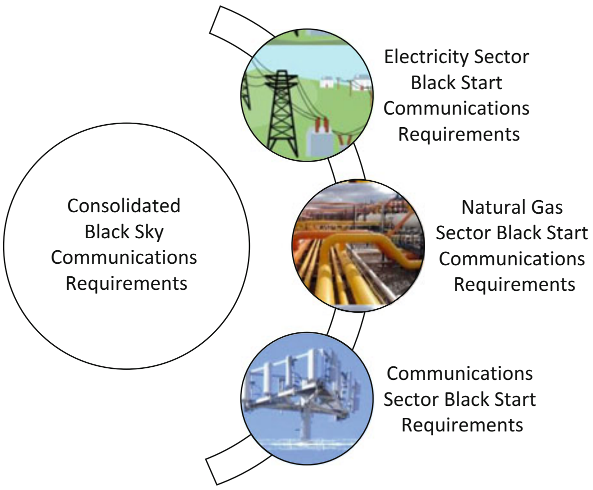

Consolidated Black Sky Communications Requirements
Electricity Sector Black Start Communications Requirements
Natural Gas Sector Black Start Communications Requirements
Communications Sector Black Start Requirements

Fig. 68.7 Convergence of black sky requirements from key sectors

68.3 Candidate Solution, Arising from the Systems Engineering Studies

Given the scope and scale of the challenge, the key issues and required information from each sector should be compiled in order to advise a systems engineering process to determine the minimum service levels required to function in a Black Sky environment. Each sector must use common planning frameworks with overlapping objectives designed specifically to support black start activities and their resulting communications requirement. The current status of compatible planning varies widely between industries relative to their degree of fragmentation, regulation and anti-trust-based collaboration. However, the comparatively limited scope of information requirements defined in this paper allows a prioritized approach to collection and gives operational benefit to all participating entities. Enabled with this information, the existing work on a systems architecture [6] can be refined to meet these sector-specific internal and external requirements (Fig. 68.7).

68.4 Summary, Portions of the Problem Remaining Unsolved, Candidate Next Steps

The interdependencies described are not simply trends that can be quickly reversed, and the services provided by each sector cannot be replicated by another to eliminate or reduce the dependency (for example, power companies cannot be expected to replace the full capability of the commercial communications sector). The cost constraints and the practicality of such an effort make it infeasible.

The intersection of these issues rests with the ability of the electricity industry to continue performing their essential functions without access to the fully-functioning versions of the system they rely on.

Several questions must be answered in order to progress towards a viable solution:

- A system-wide assessment of the communications end points needed in order to allow the electrical grid to function at a high enough level to facilitate black start operations. This effort is underway with the initial social architecture developed by Dr Siegel [7], but will need to include further study on the internal systems of grid operators. In addition, a similar assessment is needed for the natural gas industry.
- Determining the minimum amount of voice connectivity and data bandwidth needed by the electricity and natural gas industries to accomplish the black start objectives is described.
- Additional research is needed to identify the most effective hardening measures to protect back-up communication systems and its components from EMP for inclusion as part of the solution.

Enabled with the answers to these questions, technical solutions can be developed and field tested for viability and industry will need to incorporate them into their communications protocols to ensure interoperability and develop the requisite operational and emergency procedures to employ these solutions.

About the Author

Jonathon Monken is the Senior Director, System Resiliency and Strategic Coordination for PJM Interconnection, working in the areas of business continuity, physical and cyber security, risk management and resilience planning for the world's largest wholesale energy market. Most recently, Mr Monken served as VP, U.S. Operations for the EIS Council, developing best practices to improve the resilience of life-support infrastructure systems to 'Black Sky' events. Mr Monken served as Director of the Illinois Emergency Management Agency (IEMA) where he oversaw Illinois' disaster preparedness and response, nuclear safety and homeland security programs. Monken also served for 2 years as Acting Director of the Illinois State Police and possesses a distinguished military career as an armour officer for one tour of duty in Kosovo and two combat tours in Iraq, during which he was awarded the Bronze Star Medal and the Army Commendation Medal with 'V' Device for valor in combat. Monken graduated from the U.S. Military Academy at West Point, and holds an MBA from Northwestern University's Kellogg School of Management.

References

1. Situation report #13, November 3rd, 2012. U.S. Department of Energy. http://www.oe.netl.doe.gov/docs/2012_SitRep13_Sandy_11032012_300PM_v_2.pdf
2. “Using systems engineering to create a survivable communications system that will operate in the presence of “Black Sky” hazards,” by Neil Siegel, Ph.D, 1 February, 2017
3. “ A systems integration for framework for interdisciplinary black sky operations,” by Ellie Graeden and Joel Thomas, 1 February, 2017
4. Some of the preliminary results of this study are available at http://www.eiscouncil.com/Library; see the item “BSX White Paper” on that web page
5. “Using systems engineering to create a survivable communications system that will operate in the presence of “Black Sky” hazards,” by Neil Siegel, Ph.D, 1 February, 2017
6. “Emergency Communications System (ECOM), A Technical Report for the Electric Infrastructure Security Council”, by Neil Siegel, Ph.D. and Bran Ferren, 8 August 2016
7. “Emergency Communications System (ECOM), A Technical Report for the Electric Infrastructure Security Council”, by Neil Siegel, Ph.D. and Bran Ferren, 8 August 2016

Chapter 69
Black Sky Hazards: Systems Engineering as a Unique Tool to Prevent National Catastrophe

Avi Schnurr

Abstract Growing risks to our tightly interwoven infrastructures, on an unprecedented scale, have put national continuity, and our lives, in jeopardy. There is now a unique opportunity to meet this challenge.

Emerging “Black Sky” hazards ranging from large-scale cyber or electromagnetic-pulse attacks to extreme space weather or regional earthquake zones could cause subcontinent-scale power outages – a new class of extreme risks to the United States and the industrialized world. Water, fuel, and the other utilities our lives depend on cannot function without electricity, and cascading infrastructure failures would disrupt the resources essential for power grid restoration and restart, amplifying and expanding the outage and the cascading failures.

If we cannot find a solution, the first of these extreme hazards to strike the United States would be devastating. It would mean the end of our society, as we know it today.

Over the last five decades, U.S. aerospace companies invented, developed, and refined a powerful new methodology to found our high-tech world. Smoothly integrating the hyper-complexities of diverse disciplines into the dependable high-tech tools and devices we now take for granted, systems engineering has become the fundamental organizing principle used to integrate and transform disparate technologies into satellites, aircraft carriers, and spaceships.

As the new Administration in Washington D.C. begins detailing its ambitious goals for securing the nation’s infrastructures, understanding the Black Sky resilience shortfalls and interdependencies putting our lifeline infrastructures and national continuity at risk will be essential. To accomplish this, and look to industry to build the coordinated solutions that will be essential, systems engineering will be an indispensable tool.

And the work has already begun.

A. Schnurr (✉)
Electric Infrastructure Security Council, Washington, DC, USA
e-mail: aschnurr@eiscounci.org

A.M. Madni et al. (eds.), *Disciplinary Convergence in Systems Engineering Research*, https://doi.org/10.1007/978-3-319-62217-0_69

Keywords Black Sky hazards • Lifeline infrastructure • Electric interconnections • Power restoration • Black start system • Emergency response • National continuity • Training exercise

69.1 Introduction

Fueled by today's rapidly accelerating technology, with growing efficiencies derived from ever-expanding interconnectedness, modern societies depend completely on the continuity of reliable, interdependent, national-scale infrastructure systems. They have become the bedrock on which our cultures, our economies, and our lives are built. However, organically interconnected systems, while capable of superb efficiency, are subject to risks of system-wide failure from hazards that can cause comparatively small subsystem disruption. A library system missing books can still function as a library. A biological organism missing a vital organ will die.

Emerging, extreme "Black Sky" risks, including both malicious threats – electromagnetic pulse (EMP), cyber and coordinated physical attacks on critical infrastructure – and extreme terrestrial or space weather and multistate earthquake zones, could choke off the flow of critical goods and services that keeps our societal organism alive. To resolve this problem, we will need to correct the two key sources of our vulnerability to these extreme hazards: Black Sky resilience shortfalls and infrastructure interdependencies.

Systems engineering methods, adapted for these unique challenges, will be essential, and new NGO and industry-led initiatives are already laying the groundwork for this vital process.

69.2 The Role of Infrastructure Systems in Modern Nations

In the modern world, the infrastructures that underlie our lives have become so ubiquitous and effective that most of us, most of the time, simply ignore them. Evolving from centuries of urban experimentation into the extraordinarily powerful, interactive networks that we depend on today, these systems have made possible massive technology-based societies undreamed of throughout history. Yet the days when we could afford the luxury of taking these systems for granted – if they were ever truly there – have passed.

The same technology-based interconnectivity that makes possible today's megacities and our exceptionally complex culture has come with a price: the flip side of interconnectivity is interdependency.

As they become ever more interconnected and effective, our utilities and other infrastructures may increasingly be viewed as an organism: A tightly interlocked

Fig. 69.1 Watermills became an essential element of food infrastructure and an important target for invaders in the ancient world

system of systems that efficiently support, and depend upon, each other. Powerful, highly effective, and yet – like any organism – the continuity of the whole depends on the health of each of its parts. A hazard that could seriously disrupt or halt the functioning of any one of these systems on subcontinental scales immediately becomes a threat to the continuity of them all, putting at risk the survival of the organism.

And in this case, the organism is us (Fig. 69.1).

69.2.1 Historical Overview

From the earliest days of prehistory, the story of the growth of civilizations may actually be seen as the story of the development, management, and processing of natural resources – of what we call, today, lifeline infrastructure systems. The story of the collapse of civilizations, similarly, may be seen as the story of the *loss* of these resources, through either natural or malicious forces. Indeed, as I write these words, Syrian rebels have cut off the capital's primary water source, in a new escalation of the battle for that war-torn country [1].

In the immortal words of George Santayana, "Those who cannot remember the past are condemned to repeat it." Our infrastructures are the bedrock, the foundation on which all societies have been built. If we are to imagine the future of our society with sufficient clarity to understand the risks inherent in its foundation, we must begin by examining the historic origins of that foundation.

For most of history, the geographic origins of key resources and processes, throughout most of the world, were primarily local or regional. Even when social groupings grew larger, their most critical resources – their lifeline infrastructures – were generally still geographically bounded, and either resourced or stored locally. When one city's infrastructures failed from famine, siege, or other factors, the collapse of that city generally did not represent a catastrophe for its locally resourced neighbors. In fact, when empires did develop, from the time of ancient Rome through the days of the British Empire, expanded dependence on imports became a critical weak point, exploited by enemies who strived to deprive the regime of those resources, or of control over them.

Without its foundation, a house, or a nation, will fall.

69.2.2 The Magic of Modern Infrastructures: An Interconnected System of Systems

Today's integrated infrastructures supply the products and services that make modern life possible, providing electricity, food, fuel, water, pharmaceuticals, communications, transportation, and all the other resources we use in our towns, cities, factories, and every element of civil society, and most of the requirements of the nation's security establishment. And increasingly, the lifeline infrastructures that sustain our lives and our culture represent a departure from the local resource model.

In the United States, all lifeline infrastructures are tied into one of three subcontinent-scale electric "interconnections," which have become essential to the continuity of them all. Natural gas, provided as a just-in-time fuel for an increasing portion of these enormous power grids, is delivered through vast interstate pipeline networks that require continuous access to electricity to operate. All of the resources we depend upon come into existence, and are delivered to us, as components of a remarkably complex, highly integrated, and many-layered webwork of marketing, communication, production, storage, regulation, security, distribution, and finance.

Beginning with farmers, factories, manufacturers, oil fields, generating stations, water treatment facilities, and a long list of others, products and skills are provided, extracted, purified, manufactured, packaged, and turned into usable commodities and services. These commodities and services are tracked by information management systems, connected to the customers who need them through marketing and communication systems, paid for by financial systems and delivered by electrical transmission lines, pipes, the Internet, and transport by air, sea, rail, and road to storage facilities that are refrigerated, heated, humidity or pressure-controlled, and secured, or directly to countless homes, manufacturers, exporters, relief workers, retailers, and all the other economic categories which characterize modern society (Fig. 69.2).

Fig. 69.2 George Washington Bridge, connecting Manhattan with New Jersey. About 4 million people live and work on the island

Late one night recently, I found myself crossing the George Washington Bridge, a double-decked suspension bridge connecting Manhattan with New Jersey. Leaving Manhattan at that hour, I was startled to see all the incoming lanes gridlocked, filled almost exclusively with delivery trucks, bumper to bumper all across the bridge and into New Jersey. I was witnessing a small portion of the miles-long, ad hoc transportation system that magically assembles itself every night, entering Manhattan through bridges and tunnels in the final delivery stage for the infrastructures that will keep the city alive and productive for another day.

If the city is to be sustained through a wide area continuing power outage, preplanning to replicate a portion of this massive, market-driven, autonomous transportation and delivery system will be essential.

Spanning the nation and, to a large extent, the world, the complexity of our interconnected infrastructures is breath-taking; and the underlying risk is, at the very least, sobering. The magic inherent in a parent's ability, on a hot summer day in the middle of Manhattan, to grant his child's wish for a three-scoop strawberry-chocolate-vanilla cone with sprinkles is pure Harry Potter. The implications – if one imagines the complexity of the systems that meshed to produce it and must, in large part, be replicated – are terrifying (Fig. 69.3).

Fig. 69.3 A power utility "bucket truck" is loaded onto a U.S. Air Force C5 during Superstorm Sandy (*Courtesy: USAF*)

69.3 "Black Sky" Hazards: Emerging National Scale Threats to Lifeline Infrastructures

The U.S. energy sector has made great strides following recent severe weather events, increasing our national preparedness for similar events in the future. These improvements represent an excellent foundation for progress. Water utilities, the pharmaceutical industry, the emergency management community, and many other sectors have also responded to lessons learned from recent disasters.

However, in recent years, concerns have grown over a set of emerging extreme threats to the national power grid and, through cascading failures, to all other societal infrastructures. This new threat category is typically referred to as "Black Sky" to distinguish it from the "Blue Sky" or "Grey Sky" hazards that represent the more localized or short duration power outages the United States has experienced in modern times.

Black Sky hazards could cause subcontinent-scale, multiple week or month outages. They have become a growing focus of concern to State and Federal government agencies, to infrastructure sectors and an expanding set of other public and private sector stakeholders. Malicious threats, from a cyber or EMP strike to coordinated physical assault on key power grid nodes, and natural hazards, from extreme space or terrestrial weather to a newly discovered eight-state extreme earthquake zone [2], could cause multistate power outages associated with widely distributed damage that would continue for many weeks or months.

69.3.1 Societal Impact

State and federal emergency management agencies, working with the Red Cross, the Salvation Army, and the many other organizations that make up the large and highly effective mass care NGO community, have demonstrated a remarkable ability to save and sustain lives in the aftermath of the increasingly frequent, serious weather events that have plagued the nation. Katrina, Superstorm Sandy, and similar disasters have stressed, but not broken, the capabilities demonstrated by the dedicated volunteers and professionals who work to rescue their countrymen when disasters strike. And yet, without new, cost-effective, multisector resilience investment and operational planning, these first responders will find themselves unable to respond meaningfully to Black Sky hazards.

69.3.1.1 "Grey Sky" Events: National Readiness for Severe Regional or Local Weather

In the aftermath of major disasters, emergency managers and first responders have typically worked in parallel with power restoration crews and utility teams, providing shelter and care that enable the population to ride through the crisis until utilities are restored.

The key has always been infrastructure restoration, beginning with power: keeping the length of the crisis manageable for critical facilities, and for most of the affected population. When disasters have grown beyond expected levels, restoration timelines have sometimes been stretched to the limit. But even for Superstorm Sandy, with damage primarily downed power lines and distribution system failures in geographically bounded regions, power companies were able to play to their restoration strengths. Utilities used Mutual Assistance agreements to swell restoration teams with an army of linemen and bucket trucks from outside the outage zone, with the Pentagon pitching in to help. The lion's share of the power came back on within a few days.

Water and wastewater systems were close to shutting down in some areas during Sandy but, often through truly heroic efforts, water utilities were able to bridge the gap until the power came back on. Water and sewer systems continued to function, preventing what would otherwise have become an unplanned and tragic mass migration. Catastrophe was avoided.

69.3.1.2 Black Sky Events: National Readiness for Extreme, Long Duration Power Outages

While years of experience have gone into Grey Sky resilience and restoration planning, the far more comprehensive cross-sector planning needed for Black Sky hazards is just beginning. If a Black Sky event were to take place before such plans are in place, the consequences for the nation would be catastrophic.

In a Black Sky event, with the outage region covering a substantial portion of North America, bringing in an army of linemen would be neither feasible nor sufficient. The primary restoration need for these unprecedented events will be for engineering teams, and engineering support is limited today even for normal, blue sky conditions. Given today's readiness and planning levels, with insufficient engineering teams faced with extraordinary challenges in a highly disrupted environment, with seriously inadequate supplies and support, the outage will last far beyond the hours-to-days duration that has typified most outages.

Few water utilities can deal with long outages. And with water and sewage service shut down to cities in a wide swathe of the country, people will be unable to remain in their homes. A heartbreaking, unplanned mass migration would begin – with catastrophic consequences.

Why are Black Sky events so fundamentally different? Why will power restoration timelines push out beyond anything we have experienced in the past?

Some of the unique challenges for Black Sky hazards are reviewed below for the electric subsector. While the answers to these questions vary for each lifeline infrastructure, examples for the power grid can illustrate many fundamental challenges, unique to these extreme hazards, which are similar across all sectors.

- System resilience

Over many years, familiarity with the disasters that damage the power grid, varying by region and season, has led to growing investment in cost effective resilience measures. In flood zones, resilience investments often focus on raising high power relays above storm-surge levels. In hurricane country, distribution systems are moving partly underground or shifting from wooden to concrete power poles.

Without a steady, if occasional, diet of disasters, Black Sky resilience by 20–20 hindsight is not a possibility. And yet, if we expect power system engineers to have an effective starting point for restoration, a critical backbone of the power grid, region by region, will need resilience investment to protect it against the full set of Black Sky hazards.

The scope of hardware to be protected can, of course, vary from region to region depending on available resources, with trade-offs made between the complexity and duration of Black Sky restoration, and the up-front cost of resilience measures. But at least some portion of core grid restart hardware (referred to technically as "black start" and "black start cranking path" elements) will need broad protection. Ensuring that a regionally varying "backbone" of the grid is protected against EMP, cyber, physical assault, extreme terrestrial and space weather and severe earthquakes will give engineering teams the starting point they will need for restoration.

And while the details vary for the oil and natural gas subsector, the water sector, the food and pharmaceutical sectors, and each of the others, the fundamental principle is the same: Resilience investments against conventional hazards, made over time by each of the lifeline infrastructure sectors, have provided a basis for restoration against these familiar hazards. If, as a nation, we want to assure national continuity for ourselves and for future generations, such investment will need to be expanded to include Black Sky hazards, without the "help" that comes from being periodically struck by floods, tornados, and hurricanes.

- Technical support

Power system engineers are in short supply even on normal “Blue Sky” days. In disrupted, Black Sky scenarios, without prearranged family support for restoration teams, many engineers may simply be unable to report to work, further reducing the availability of technical personnel.

Given adequate coordination with mass care NGOs [3], this concern could be addressed in advance. However, with damage assessment required on a subcontinental scale, finding damaged components among thousands of generating stations, tens of thousands of power substations and a wide variety of other grid hardware would be an enormously complex process even under ordinary conditions. For Black Sky scenarios, supplemental, prearranged, precertified technical support will be crucial [4]. New technological approaches, including remotely reporting grid health diagnostics, could also be highly leveraging. With a potential for paying synergistic operational dividends on Blue Sky days, such approaches may be well worth exploring.

- Transportation

With huge areas to cover and limited available personnel, even the first stage in power restoration – locating damaged components – will take far more than the hours or days we have come to expect. Roads will be gridlocked by migrating residents, gas stations inoperative, and food and potable water generally unavailable. To find the damage, without focused, preplanned transportation and security support from National Guard personnel [5] or other resources, engineering teams will struggle to get from site to site. And as they locate damage, bringing repair and replacement components in – especially for very large, heavy units like generator components and extra high voltage (EHV) transformers – will be difficult and painfully slow, at best (Fig. 69.4).

Fig. 69.4 In the aftermath of any Black Sky event, major highways will be gridlocked

- Deployed spares

Once Black Sky resilience investment and coordinated planning by the Electric and the Oil and Natural Gas Subsectors reaches an adequate Black Sky threshold, there would never be a complete Black Sky power outage. Region by region, at least some portions of the power grid would either continue to operate or quickly restart, regardless of the nature of a Black Sky event, or its severity.

Starting from that point, engineers would then need to find damaged components, and locate and acquire the necessary spares. This will be far from simple, with transportation disrupted, and with the normal processes of finding and acquiring spares (from, for example, online suppliers) impossible. Importantly, this will be the case even when the components involved may be otherwise insignificant and extremely inexpensive – components whose availability, under normal circumstances, is taken for granted.

For Black Sky scenarios, carefully devised spare inventories associated with anticipated categories of damage for each of these scenarios will need to be predeployed, at the sites where they may be needed. And where vulnerable components are too expensive or too large to conveniently predeploy, operational planning will be needed to widely communicate location and availability, with transportation precoordinated with National Guard and security personnel.

Such an approach, with its requisite thorough, detailed advance planning, will typically also be applicable to other lifeline infrastructure sectors, which will likely face similar challenges.

- High value, long-lead hardware

Black Sky hazards involve malicious or natural scenarios that will cause damage to a range of conventional grid components. However, large high voltage systems such as generators or EHV transformers will also be at risk. Given the many months or even years it generally takes, under optimal conditions, to acquire such components, failure to protect them would mean that, if damaged in significant numbers, they will be basically unreplaceable. There could then be no meaningful restart for the affected portions of the power grid.

As a key element of ensuring Black Sky resilience, developing, validating and implementing affordable, cost effective approaches to protect such hardware against each of the Black Sky hazards will be essential. For water and wastewater, oil and natural gas, food, pharmaceutical, and other lifeline infrastructure subsectors, this same process is equally applicable. Identifying critical high value, long-lead hardware and finding and broadly implementing cost-effective protection will be essential to ensure timely Black Sky restoration.

- Communications

In an ongoing, long duration massive power outage, both land-line and cellular telephone networks will be down. Satellite phone systems will be overwhelmed and jammed, if operating. While many companies and agencies have emergency communication systems for their own facilities, continued availability and operation of such systems will be increasingly at risk in a long duration outage, and successful power restoration in disrupted Black Sky scenarios will depend on support from – and communication with – a wide spectrum of public and private stakeholders.

A predeployed, widely distributed emergency communication system will be essential for these extreme scenarios. Each node for such a system will need its own long-duration power supply, and embedded protection against the full set of Black Sky hazards. To enable both multisector restoration support and saving and sustaining lives, the system will need to be fully interoperable with any existing emergency communication systems that may continue to operate, capable of connecting all elements of the nation's lifeline infrastructures, and tying in a wide array of private sector asset owners and government agencies [6].

Without such a system, power system engineering teams – like their colleagues in other infrastructure sectors – would find their efforts impeded the first time they need to reach out for external support or coordination.

- Interdependencies

When the geographic boundaries and duration of a power outage exceed conventional disaster thresholds, without careful, thoroughly coordinated and operationally focused multisector planning, there would be cascading failures of all of the interdependent lifeline infrastructure sectors. If we look at these closely interconnected sectors as an integrated, organic system, the Electricity Subsector would be analogous to a circulatory system. As for a living organism, society's interdependent infrastructure network can be sustained for only a very brief duration without it, and yet the power grid's growing fleet of gas-fired generators will only operate when they receive just-in-time fuel delivered through pipelines that, in turn, require their electricity to function.

Resolving this problem, bridging a gap of weeks or months without the normal autonomous webwork of interconnectivity that keeps the entire system of infrastructure systems healthy, will only be possible if a mechanism is found to host careful, thoroughly coordinated resilience investment decision-making and operational planning; an approach that can overcome the "stove-piped" limitations that often beset broad planning initiatives.

Without such planning, what would be the impact of the many cross-sector interdependencies?

Based on recent research [7] focused on Black Sky scenarios, two important examples may be helpful.

(a) *Electric Subsector/Oil and Natural Gas Subsector/NGO Sector/Transportation Sector/State Sector/Communication Sector Interdependencies:* With continuing expansion in use of natural gas as the fuel of choice for a rapidly growing, regionally varying segment of the nation's electricity generating stations, the electricity/natural gas nexus has become a subject of critical, cross-subsector concern. Gas is provided as a just-in-time fuel to generators, generally transported from gathering fields via long interstate gas lines. Yet, the pumping stations that power these pipelines require electricity. If the power grid goes down for a duration longer than a few days, emergency generators at these pipeline pumping stations will stop, halting the flow of gas and depriving all generating stations using a pipeline of fuel. In such an emergency, getting fresh diesel fuel to pipeline pumping stations would be a priority. However, without extensive cross-sector preplanning by State Emergency Managers, National Guard, mass care NGOs, and transportation stakeholders, and some provision for Black Sky emergency communications, this will not be feasible. Diesel trucking companies in the outage area will generally not be operating in this scenario. Their offices will be nonfunctional; their telephones will be out of service; roads gridlocked; gas stations for truck refueling inoperative; food, shelter, and security unavailable along routes; and GPS navigation systems inoperative.

(b) *Electric Subsector/Water Sector/Chemical Processing Sector/Transportation Sector/Emergency Management Sector/Communication Sector Interdependencies*: Many, though not all water companies maintain emergency generators on many of the large pumps, or "lifts," that bring the water from aquifers or other sources to treatment facilities. Some also have emergency generators available for treatment facilities. While options like reducing pressure and treatment levels could reduce their emergency generation requirements, water companies have not planned or implemented such fall-back capabilities. Those companies who maintain emergency generators rarely have provisions for more than a day or so of diesel fuel, and their emergency generators are not designed to be EMP protected, or to operate continuously for long durations. Treatment facilities need chemicals, and yet, without a functional emergency communication system, requesting fresh chemicals will not be possible, the companies providing these chemicals will typically not be functioning, and their delivery trucks would be dealing with disrupted conditions on public roads (Fig. 69.5).

These two examples are representative of a wide array of similar cross-sector interdependencies. If they are not resolved, utilities and infrastructure sectors will be able to restart and restore operations only if all their partner sectors are already functioning. This classic, deadly "Catch 22" situation would make meaningful restoration of the power grid and the other, power-dependent utilities impossible

Fig. 69.5 Water utility pumping station

within time frames that could prevent tragic, unplanned mass migration, and the death of millions.

As a nation, we cannot afford to allow that to happen. What can be done?

The challenge of finding an effective, operationally oriented mechanism to integrate highly unique, specific plans across many disparate sectors is, fortunately, not new. In the aerospace and engineering worlds, this challenge is faced every time a team begins a contract that calls for conceiving, designing, building, and testing a new and complex system. Their starting point: systems engineering.

While applying systems engineering methodologies to carefully integrate corporate, NGO and government agency planning may be original, there is no reason, in concept, that it should not be possible. Indeed, as the best-in-class discipline available for developing carefully coordinated complex systems, systems engineering methodologies will be crucial if the nation is to define the focused resilience investments and develop the operational planning which, alone, can ensure national continuity that will bridge the first Black Sky gap we encounter.

The next section illustrates a new systems engineering initiative for coordinated planning, already being utilized by industry and government leaders in an ongoing, hosted planning process.

69.4 Systems Engineering Methodologies: A Template for Addressing Black Sky Hazards

The origins of the term *systems engineering*, and its first use, began at Bell Laboratories in the middle of the last century [8]. As large systems of unprecedented complexity were produced, especially for the U.S. military, aerospace companies developed and used the methodology to ensure a well-coordinated process relating all the interdependent elements of a system to each other, and to keep them focused effectively on overall system goals. It quickly became the best-in-class discipline for developing large and intricate systems.

Given the extraordinary scale and complexity of the United States' many lifeline infrastructure systems, systems engineering is ideally suited to framing the smoothly orchestrated planning needed to help infrastructure sectors lay out coordinated paths forward – to build plans that can bridge the national resilience gap for Black Sky hazards.

This approach is now underway, being led by NGOs and industry as they begin addressing this urgent need. It is at the core of a new initiative, the Electric Infrastructure Protection (EPRO®) SECTOR Project. Facilitated and hosted by EIS Council for an expanding set of infrastructure sectors and their government, NGO and corporate partners, recommendations for Black Sky resilience and restoration support planning are being developed by leading executives and operational managers within each sector. As a platform for development and dissemination of their work, the recommendations emerging from each sector are documented in

EPRO Black Sky Playbooks that embody key features of the classic systems engineering process.

An application of traditional systems engineering, EPRO SECTOR, has been developed as a platform to foster carefully coordinated, multisector planning, with synergistic sector objectives focused on a common, overall mission.

69.4.1 Overall Black Sky Mission

As in any systems engineering process, the first step is to describe the top level mission goals which characterize and define success. For Black Sky resilience, this mission becomes the common goal or objective that every subsystem – in this case, every sector – will work to support.

69.4.2 Sector Service Levels or Priorities

Given the overall Black Sky mission, each sector can use this as a focus to review its normal performance or restoration priorities, and develop options for unique, sustainable performance levels, or restoration priorities, that optimize the sector's ability to support that top level mission. Thus, for example, water sector managers have developed options for reduced water pressure and treatment levels, lowering their electricity and treatment chemical requirements, and thus expanding the time a utility can sustain reduced operations without replacement fuel or chemicals.

69.4.3 Sector Internal Requirements

With a menu of options for reduced service levels or revised restoration priorities defined, the next step is to develop specific, recommended approaches utilities can take which will accommodate these options, with associated resilience investments and new operational planning. For water utilities, for example, this could involve providing hardware provisions and operational planning to reduce pump pressure or shut down service to some regions, and allow treatment facilities to shift to an emergency mode providing potable water without addressing other, less critical treatment requirements.

An Important Note on Setting Resilience Goals For these internal requirements to fully address the nation's resilience gap for Black Sky hazards, it is crucial that *robust goals* be set for each sector's resilience investments. By focusing on developing affordable, cost effective measures that can provide the requisite protection against the "high end" of projected possible impacts for each Black Sky hazard, the years of effort ahead to achieve those goals will ensure national continuity. If sectors set timid goals, either by working to address only the "weakest end" of possible impacts, or by ignoring the urgency of finding cost effective solutions, those years of effort could be wasted. And the nation's future lost.

69.4.4 Sector External Requirements

Recommended Black Sky resilience investments and operational plans will allow each sector to optimize its own performance, to increase its ability to sustain service at some predefined level, or to plan for Black Sky-optimized restoration. These investments and changes, sector by sector, will maximize service or restoration within a sector. However, since each sector will have only a limited capability to sustain operations or pursue restoration efforts on its own, these internal changes will need to be supplemented in critical areas, determined by each sector, where a partner's support will be essential.

Focused on the products, services, or regulatory relief each sector will need from its utility, NGO, and government partners, as a final step in this process, these external requirements (sometimes known as interface requirements) are made available to those partners. The sectors then each review these "external requirements," allocated by other sectors. If acceptable, provisions for meeting them become part of the appropriate sector's internal requirements. If they exceed a sector's estimation of its own Black Sky capabilities, they become the subject of a negotiation process, pursued either in cross-sector working group meetings or as part of periodic, multisector EPRO SECTOR Executive Committee meetings.

69.4.5 Common, Cross-Sector Requirements

With different and unique roles and structure for each infrastructure sector, and for their partners in government and in the mass care NGO community, each sector is developing unique plans for Black Sky service levels or priorities, with associated internal and external requirements. However, some capabilities or hosted services will be needed by all sectors. In these cases, each sector is working to define their needs and opportunities associated with these common, cross-sector capabilities. Two different initiatives are now moving forward to address these multisector needs (Fig. 69.6).

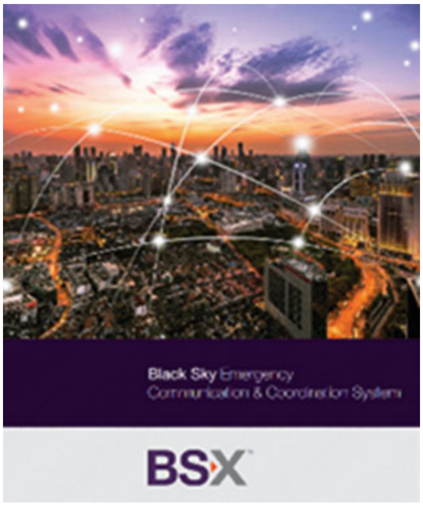


Fig. 69.6 The BSX Whitepaper reviews the architecture requirements for Black Sky emergency communication and coordination

69.4.5.1 Black Sky Communication and Coordination System™ (BSX™)

A primary ground rule essential for any hope of infrastructure restoration and recovery following a Black subcontinent-scale outage is a low cost, basic, Black Sky compatible, nationally deployed emergency communication system, designed to support at least 100,000 nodes. To be effective in these scenarios, each node in the system must meet its own power requirements for a month or more, and the system must provide and manage its own communication paths and linkages (with no need for leveraging the nation's existing telecommunications infrastructures). The system's architecture must also be designed to support very wide, multisector use, to allow operations personnel in infrastructure sectors throughout the United States to communicate with their colleagues in other infrastructures, with corporate asset owners, service providers, government agencies, and NGOs. And given the inevitable complexity of real time coordination for these severely disrupted scenarios, the system must also support, and provide for, embedded situational awareness and decision support.

The BSX™ system initiative represents a primary example of a system architecture meeting these requirements [9].

69.4.5.2 Emergency All-Sector Response, Transnational Hazard Exercise™ (EARTH EX™)

Comprehensive planning is critical for preparedness in response to any emergency. For conventional hazards, corporations, volunteer organizations, government agencies, and others involved in emergency response or infrastructure restoration use regular exercises and training to ensure their response teams will be properly prepared. Black Sky events will share a similar need, with one crucial difference: the most complex aspect of restoration support and emergency response planning for these unique scenarios is the need for thoroughly integrated cross-sector coordination. For this hazard category, for exercises to be effective, they must include joint participation of, as many as possible, those sectors that will need to work smoothly together, to validate their coordinated processes, interoperable equipment, and other aspects of integrated planning.

The EARTH EX initiative, beginning in 2017, represents an initial step toward building such coordinated Black Sky exercise planning.

69.5 Summary

Never before in history have the infrastructures required to sustain life been subcontinental in scale.

Never before in history have all lifeline infrastructures been fully interdependent.

We have gained extraordinary power over our lives and our world by building a vast societal organism, almost biological in the interconnectedness and complexity of its essential utilities and resources. But, as we are now discovering, this has come at a price.

Modern nations that do not recognize the unique vulnerability of these tightly interlinked systems, or wait for 20–20 hindsight to understand the potential severity of Black Sky hazards, will only continue until the first extreme hazard strikes. Thereafter, struggling through tragedy on an unprecedented scale, survivors will be forced to deal with an existential nightmare.

We now have an excellent opportunity to set out on a very different path by setting robust goals for affordable resilience and infrastructure restoration that address the most severe projections for emerging Black Sky hazards. Systems engineering, the powerful discipline used routinely by the aerospace industry for coordinated development of complex systems, provides an outstanding, best-in-class tool to address these goals.

A new systems engineering-based Black Sky planning process is now moving forward, initiated and led by NGOs and industry. In addition, as the new Administration in Washington D.C. develops its infrastructure investment initiatives, there is also a unique opportunity for this process to add value to government planning that can enhance and protect our nation's critical resources.

With dedication, diligence, and hard work, we can secure our birthright for ourselves, for our children, and for future generations.

References

1. News reference, as Syrian rebels cut off the primary water supply for Damascus: http://www.ynetnews.com/articles/0,7340,L-4899137,00.html
2. For more information on the CUSEC New Madrid Seismic Zone Catastrophic Planning Project, visit: http://www.cusec.org/plans-a-programs/multi-state-planning/156-cusec-new-madrid-seismic-zone-catastrophic-planning-project.html
3. For a review of possible approaches, such as prioritizing NGO shelters and support to zones where the families of restoration personnel are most heavily concentrated, see EPRO Handbook I, Ch. 1, Sec. III, pp. 80–86, and Ch. 3, Sec. IV, 227–232; http://www.eiscouncil.org/App_Data/Upload/3dadf58f-7457-46bf-92a4-551c6608d925.pdf
4. See, for example, the Certified Power Restoration (CPR^{TM}) Initiative, as a mechanism to preplan, before a Black Sky hazard, availability of substantial, supplemental certified technical support personnel: http://www.eiscouncil.org/App_Data/Upload/CPR%20ET%20Whitepaper%20(1).pdf
5. For considerations of possible roles for National Guard, DOD or other security personnel in addressing critical transportation needs for power restoration, see EPRO Handbook I, Ch. 3, sections. V and VI, pp. 233–253; http://www.eiscouncil.org/App_Data/Upload/3dadf58f-7457-46bf-92a4-551c6608d925.pdf
6. For an example of a widely deployed emergency communication system architecture, designed to address Black Sky scenarios, see the Black Sky Emergency Communication and Coordination (BSX^{TM}) System whitepaper, at: http://www.eiscouncil.org/App_Data/Upload/50ba2a93-adef-465f-86ef-caef6e6cf8c8.pdf
7. Extensive work has been done recently to review, in particular, interdependencies among the electricity and oil and natural gas subsectors, and the water sectors. This original research (principal author and editor, Dr. Paul Stockton), was compiled and published recently in EPRO Handbook II, with its two volumes – EPRO Handbook II – Volume 1 (Fuels) and EPRO Handbook II – Volume 2 (Water). To download softcopies, visit EIS Council's online library, at: http://www.eiscouncil.org/Library. To request hardcopies, write to info@eiscouncil.org
8. For historical information on the origins of systems engineering, see Schlager, J. (July 1956). Systems engineering: key to modern development. IRE Transactions. EM–3 (3): 64–66. doi:10.1109/IRET-EM.1956.5007383
9. To review the BSX whitepaper, visit: http://www.eiscouncil.org/App_Data/Upload/50ba2a93-adef-465f-86ef-caef6e6cf8c8.pdf

Chapter 70
Agile Fit Check Framework for Government Acquisition Programs

Supannika K. Mobasser

Abstract To embrace the rapid rate of change in software system development, it is crucial to be flexible, resilient, and robust. Although there is no one-size-fits-all process, an agile approach has been increasingly popular and widely adopted in the software industry. Aerospace and defense contractors have gradually started to adopt agile principles and its manifesto. However, there are questions on whether agile would be a good fit for large-scale mission-critical programs or better yet how we know whether we should use agile. If so, should we use agile as it is used in commercial software-intensive industry or should we tailor the process so that it is still agile but also compliant with government regulations. This paper discusses a framework that can be used to check the fitness of a proposal for an agile development based on the system's characteristics and on both the government and the offerors' readiness. This framework does not give a yes/no answer on whether a program should use an agile development process, but assists the program office in the identification and mitigation of the risks of an agile approach for a particular system. This framework will also identify opportunities for process tailoring and improvement. The framework was developed with the proposal phase in mind; however, it can be used at any time during the software development lifecycle to help mitigate agile development risks.

Keywords Agile • Agile Fit Check • Government Acquisition • Software development • Software development lifecycle

70.1 Introduction

It is a big challenge for large-scale government projects to follow a purely agile software development approach due to constraints such as lack of close and continuous user-developer collaboration; insufficient agile knowledge within the government team; scalability impacts on size, coordination, and criticality; and difficulty for the product owner team to speak as one voice. At the beginning of the

S.K. Mobasser (✉)
The Aerospace Corporation, El Segundo, CA, USA
e-mail: supannika.k.mobasser@aero.org

project, the government team has a shared vision of how the project would look and they can speak as one voice. If a project team were to adopt the agile development approach, what evaluation criteria could they use to determine if the approach was a good fit? Would it be compatible, and would there be ramifications? To address these questions, the Agile Fit Check Framework was developed.

70.2 Agile Fit Check Framework

The Agile Fit Check Framework was developed based on the Balancing Agility and Discipline model [1], MITRE's agile assessment list [2], agile research studies, and many other sources. The framework can be used to check the compatibility of the agile development method for use on a program or project. However, it does not give a yes/no answer on whether a program should use an agile development process. The framework also identifies potential impediments of adopting an agile approach for a particular system. Agile fitness can be evaluated based on criteria in three major categories: (1) system's characteristics, (2) government's level of commitment, and (3) contractor's level of commitment.

70.2.1 System's Characteristics

Agile methods allow "fail fast, fail often," accommodate frequent requirements changes, and provide quick feedback. The projects that benefit most from an agile development are projects that have high requirements or scope volatility; ability to decompose requirements to fit small increments; systems that have the ability to accept frequent upgrades; and development environments that fully support integration and test throughout the development lifecycle. It is also generally more effective to use an agile development method on projects that are built on a mature infrastructure or architecture.

70.2.1.1 System Criticality

While agile has been popular in the software development industry, agile is still not a common practice in large-scale mission-critical system development. Although it has been shown that agile can be used in mission-critical projects, it is more prudent to apply agile to low criticality projects. The possible ramifications of using an agile development on a system of high criticality are as follows: (a) frequent changes could lead to inconsistencies in artifact maintenance and standards compliance; (b) systemic properties, such as "–ilities," may not be incorporated from day one; (c) with a design-as-you-go method, the architecture may end up being unsuitable or uninteroperable, and possibility of having requirements "fall through the cracks" on large systems; and (d) there is potentially more difficulty in managing the white space due to disjointed development groups employing design-as-you-go methods. System criticality can be categorized into five levels:

- Level 1 – Minimal impact on nonessential capabilities. Some user inconvenience if the software system fails.
- Level 2 – Minor. Affects nonessential capabilities. Work-around exists.
- Level 3 – Moderately affects essential capabilities. Work-around impacts operational productivity.
- Level 4 – Significantly affects essential capabilities. Possible work-around might cause significant delay or degraded capabilities.
- Level 5 – Severely impacts the mission because it prevents essential capabilities from being carried out. There is no work-around.

70.2.1.2 Requirements Volatility

One of the 12 principles [5] is "Welcome changing requirements, even late in development." Requirements volatility can be measured by the amount of the software requirements change or expected change in a month. The projects with high requirements volatility will greatly fit for agile adoption. While a project with high requirements volatility is difficult to accommodate in a plan-driven method that would require frequent updates to documentation and requirements traceability matrices, a project with low requirements volatility is generally beneficial for any development method. Requirements volatility can be categorized [1] into five levels:

- Level 1 – Around or at least 50% (or anticipated) change over the project life or 50 functional requirements change/month
- Level 2 – Around 30% (or anticipated) change over the project life or 35 functional requirements change/month
- Level 3 – Around 10% (or anticipated) change over the project life or 25 functional requirements change/month
- Level 4 – Around 5% (or anticipated) change over the project life or 10 functional requirements change/month
- Level 5 – Around 1% (or anticipated) change over the project life or 1 functional requirements change/month

70.2.1.3 Requirements Formality and Detail

Requirements formality and detail refers to the level of maturity of the software requirements or user-level requirements and the level of effort on the software requirements definition upfront. Agile does not favor comprehensive documentation [4], but encourages simplicity [5] or the art of maximizing the amount of work not done. As a result, a project that requires full-scale documentation upfront could impose a risk of premature commitment and potential unnecessary reworks. Well-understood and constitutional software requirements are not a risk for agile. However, upfront derivation of a full set of requirements statements and trade studies is not consistent with the principle of simplicity. The possible ramifications of following a plan-driven process on a project with low requirements derivation formality and detailed analyses include higher likelihood of misunderstanding of the requirements, higher chance of delivering a system that will not meet requirements, premature (architecture and design) commitment to unclear requirements, unknown requirements priorities, and low value product features that may get developed first, resulting in late rush to bolt on higher priority capabilities. Five levels of requirements formality and detail are

- Level 1 – Emergent software requirements, unprecedented systems, or unknown solutions. Requirements are in the form of business objectives, epics, and user stories.
- Level 2 – Many undefined areas. Many software requirements are unclear or unprecedented. Requirements are in the form of business objectives, epics, and user stories.
- Level 3 – Some upfront analysis and trade studies, some undefined areas, unclear or unprecedented requirements.
- Level 4 – Few undefined areas. Most software requirements are captured in detailed requirements statements or in detailed use case scenarios.
- Level 5 – Upfront detailed requirements statements, use case scenarios, complete detailed analysis, and trade studies. Software requirements are well understood and constitutional.

70.2.1.4 Requirements Decomposition

Agile prefers to have short time boxes that allow the team to deliver working software frequently. Therefore, the software requirements should be decomposable into small tasks that fit into short development cycles. It is also crucial to consider whether each software requirement can stand by itself or highly depend on other requirements. The possible ramifications of a project with low ability for requirements decomposition or having requirements that are too big to fit in a sprint include (a) unable to complete the requirement in a single sprint and may have to frequently adjust the timebox, which leads to breaking the sustainable pace, and (b) scalability risks – interdependent requirements have numerous dependencies that must be satisfied. The development might not be able to start testing parts of the requirement until the whole requirement is complete. Five levels of requirements decomposition are:

- Level 1 – Software requirements are decomposable into small tasks that fit a sprint. Requirements are completely independent on other requirements.
- Level 2 – 80–90% of the software requirements are decomposable to fit a sprint.
- Level 3 – 60–80% of the software requirements are decomposable to fit a sprint.

- Level 4 – 40–60% of the software requirements are decomposable to fit a sprint.
- Level 5 – Requirements are tightly coupled and completely dependent on other requirementsAgile Fit Check:.

70.2.1.5 Deployment Timelines

Deployment or release timelines refers to the length of the release duration and whether the operational environment easily accommodates frequent software updates. One of the agile principles [5] is “Deliver working software frequently with a preference to the shorter timescale.” Agile works really well with a project with aggressive deadlines or urgency. If an agile project is not able to support short releases, the project may not get timely validation or may be unable to obtain timely validation of implemented features from the users that would support fixes in a timely manner. If coupled with not doing continuous integration and deliveries, it could lead to integration challenges or system failure. Five levels of deployment timelines are:

- Level 1 – The system accommodates frequent (daily to monthly) releases into operational environment or an independent test.
- Level 2 – The system accommodates bimonthly or quarterly releases into operational environment or an independent test.
- Level 3 – The system supports semiannual to annual releases into operational environment or an independent test.
- Level 4 – The system supports periodic releases into operational environment or independent test.
- Level 5 – The system has difficulty tolerating periodic releases.

70.2.1.6 System Integration Interval

It is important for agile projects to synchronize and integrate their works in a sustainably short timescale. The long build cycle would break the agile model of delivering working software frequently, loss of opportunities for frequent feedback, loss of continuous integration and testing, and higher cost of rework if found late in the lifecycle that components are not integratable or not interoperable. Five levels of system integration interval are:

- Level 1 – Continuous integration
- Level 2 – Weekly build
- Level 3 – Monthly build
- Level 4 – Semiannual build
- Level 5 – Annual build

70.2.1.7 Program Scope

The agile process favors design-as-you-go approach, where there is no dedicated phase for upfront analysis and design. Hence, there are bigger challenges for a program that involves numerous interdependencies between several architecture layers. Although incremental design helps in avoiding premature design commitment, but in a multiowner development, the development team needs a baseline design that they can commit to. The possible ramifications include (a) lack of sufficient planning for the system-level architecture; (b) focus on rapid development that may result in a lack of system-level perspective; (c) with design-as-you-go, if the interfaces are not designed properly, interdependent components from different subsystems that may not be interoperable; (d) insufficient attention to the incorporation of system-level quality attributes; and (e) system "-ilities" that may be overlooked. Five classifications of program scope are:

- Level 1 – The development is limited to the application layer with an existing robust architecture.
- Level 2 – The development covers the application layer with an existing architecture that needs minor revision.
- Level 3 – The development covers the application layer with an existing architecture that needs major revision.
- Level 4 – The development covers the application, the new underlying architecture, and interdependent systems.
- Level 5 – The development covers the application layer, the new underlying architecture and supporting platform, and highly interdependent systems.

70.2.2 Government's Level of Commitment

The government side is a primary factor in this framework. There is a myth that with an agile project, the government team can let the development team do whatever they want. In reality, it is quite the opposite. Agile development methods require constant and colocated customer involvement. It is crucial for agile projects to have leadership support, a proper contracting strategy, frequent and effective collaboration, and good oversight tools. An agile development also requires availability of target users for feedback. Importantly, the government team needs to have sufficient knowledge of the agile process in order to set the right expectations and contribute effectively to the agile project.

70.2.2.1 Leadership Support

It is crucial to get buy-in from leadership and it is important to know how open leadership is to modification of traditional management and technical lifecycle activities and reviews to support agile methodologies. If the leadership does not support agile approach or prefers a plan-driven approach, the possible ramifications include the following: (a) there would be higher likelihood of expectation mismatch with regard to project progress; (b) there would be higher likelihood of needing additional or unnecessary plans, meetings, reviews, or control layers; and (c) program leadership may deemphasize or overturn team decisions. Five levels of leadership support are:

- Level 1 – The program office leadership fully understands and supports agile approach.
- Level 2 – The program office leadership supports agile software development.
- Level 3 – The program office leadership supports hybrid or tailored agile processes.
- Level 4 – The program office leadership is somewhat open to a nontraditional process.
- Level 5 – The program office leadership prefers a traditional development approach.

70.2.2.2 Contracting Strategy

The contract for an agile project should support the nature of agile software development such as short and incremental development cycles and continuous requirements refinement. Hence, the contract should include support for multiple short deliveries; frequent or continuous changes in project scope and requirements; minimal or no contract data requirements list (CDRL) items upfront; and government collocation (e.g., office space and resources) at developer sites or travel budget for frequent face-to-face meetings and telecommunication tools and capabilities. Five classifications of contracting strategy are:

- Level 1 – The contract supports short development timelines, incremental or modular development, requirements refinement throughout the development process, full documents and project details not required upfront, and frequent capability releases.
- Level 2 – The contract supports short development timelines, incremental or modular development, requirements refinement throughout the development process, and frequent capability release.
- Level 3 – The contract supports short development timelines, incremental or modular development, and requirements refinement throughout the development process.
- Level 4 – The contract supports short development timelines, incremental or modular development, and fixed scope and requirements.
- Level 5 – The contract requires sequential development with fixed milestone reviews and exit criteria, fixed scope and requirements, and formal documents and project details upfront.

70.2.2.3 Government Expertise

In order to provide an appropriate oversight, the government team must be familiar with an agile approach. A project that follows an agile development process but is working with a government team that does not have agile expertise/experience could cause the following impediments: (a) significant cultural change in modus operandi for the government team, (b) expectation mismatch from confusion regarding project progress, and (c) potentially high personnel turnover due to inability to adjust work styles from a plan-driven to an agile process. Five classifications of government expertise are:

- Level 1 – All program office team members received training and have experience in agile projects. Some have acted as a product owner.
- Level 2 – Most program office team members received training and have agile experience.
- Level 3 – Some program office team members have access to training or have agile experience.
- Level 4 – Some program office team members have basic agile knowledge.
- Level 5 – No program office team member has access to agile training or prior experience.

70.2.2.4 Level of Oversight

With the close and continuous collaboration between the government team and the development team, the government team should have the authority to determine the priorities of tasks during planning or make significant scope decisions such as to increase or reduce the scope of the system under development or to reduce or lengthen the schedule without going through formal review boards. With the proper level of oversight, there is a high likelihood of agile process breakdown due to impeded progress on backlog prioritization by the development team due to the slower decision-making process imposed by program office leadership. Five levels of oversight are:

- Level 1 – Program office software team has the authority to make real-time or quick turnaround significant scope (cost, schedule, technical) or development priority decisions.
- Level 2 – Program office software team has the authority to make real-time or quick turnaround significant technical or development priority decisions.
- Level 3 – Program office software team members have the authority to make limited real-time or quick turnaround technical decisions.
- Level 4 – Program office authorization is needed to make significant technical decisions.
- Level 5 – Program office authorization is needed to make all technical decisions.

70.2.2.5 Collaboration

Agile emphasizes frequent collaboration and preferably face-to-face interactions [3]. If the program office acts as a product owner, it is essential that they have to be able to provide feedback to the development team quickly. The government team should also be able to regularly support monthly reviews/demos. The possible ramifications of insufficient collaboration tools and resources are (a) higher likelihood of schedule lag resulting from delays in customer concurrence on proposed requirements changes/interpretation or backlog prioritization by the development team, (b) higher likelihood of not meeting customer or user expectations and/or requirements churn if they are not able to participate in the frequent requirements interpretations or backlog prioritization, and (c) additional effort in formalizing the communication medium. Five levels of collaboration are:

- Level 1 – Program office software team and software development team(s) are colocated and are dedicated to the project. Communication medium ranges from face-to-face daily conversations, data accession list, e-mails to as-built documentation.
- Level 2 – Program office software team members can be available on short notice for meetings or quick feedback and can commit to attend monthly review or product demo. Collaboration is done through frequent verbal conversations, data accession list, e-mails, and as-built documentation.
- Level 3 – Program office software team is generally able to provide quick feedback or attend meetings. Collaboration is done through frequent meetings, e-mails, and formal documentation.
- Level 4 – If planned ahead, program office software team is available for meetings or feedback. Collaboration is done through meetings and formal documentation.
- Level 5 – Program office software team and software development team(s) are in different time zones and unable to support frequent meetings or to provide feedback in a quick turnaround time or attend monthly reviews. Collaboration is done mainly through formal meetings and formal documentation.

70.2.2.6 End User Involvement

The first agile principle [5] is “to satisfy the customer through early and continuous delivery of valuable software.” The principle indicates that agile project requires continuous customer or end user involvement from day one, so it is important to check the degree of availability and commitment of end users or representatives throughout the entire development lifecycle. The possible ramifications of not having consistent availability or input from a user or representative are (a) higher likelihood of not meeting customer or user expectations and/or requirements churn if they are not able to participate in the requirements refinement/interpretation or specific user-related development decisions; (b) higher likelihood of user dissatisfaction due to lack of feedback on the product's usability, functionality, and performance expectations during code development; and (c) higher likelihood of cost increase due to rework. Five levels of end user involvement are:

- Level 1 – End users are readily available throughout the entire development lifecycle. End users commit to participate in the monthly reviews.
- Level 2 – End users are regularly available to provide feedback and to participate at reviews.
- Level 3 – End users are available to provide feedback or to participate at monthly reviews.
- Level 4 – Some end users are available at some monthly reviews.
- Level 5 – No target user representative or not available to provide feedback.

70.2.3 Contractor's Level of Commitment

The third key component of an agile project is the contractor or the development team. Agile development methods require a dedicated, motivated, and experienced development team. A good agile team requires an extremely high level of synchronization and quick turnaround process at a sustainable pace. Hence, an ideal agile team should be a small team with an agile-ready mindset, knowledge, and skills. Due to the high degree of collaboration and frequent re-baselining, agile development methods work best with one or a few contractors. It is also best to have a colocated team with low to zero personnel turnover and collaborative development infrastructure that supports the one-team, one-voice paradigm, for dynamic and rapid fielding.

70.2.3.1 Developer Expertise

With a quick pace and nontraditional practices of agile development, agile requires the development team and scrum master or team facilitator to be trained and have experience in agile development. The possible ramifications of not having agile-ready and agile-mindset development team are as follows: (a) agile requires significant cultural changes and commitment from all stakeholders in all levels – significant cultural change in modus operandi including adjustment to having greater active government team participation in the software design and development discussions. (b) Steep agile learning curve may require a year to build an effective agile team. (c) Agile-not ready development may cause reduction in process rigor, and (d) there is potentially high personnel turnover due to rejection of agile development processes by pro plan-driven developers. Five classifications of developer expertise are:

- Level 1 – All developers and the scrum master have prior hands-on experience in an agile software development and scrum master is certified.

- Level 2 – Most developers and the scrum master have prior hands-on experience in agile software development and scrum master is certified.
- Level 3 – Some developers and the scrum master have prior hands-on experience in agile software development. Hands-on agile coaching is available.
- Level 4 – Some developers have basic agile knowledge or hands-on agile coaching is available.
- Level 5 – None of the developers have agile knowledge or have participated in an agile project, and hands-on agile coaching is unavailable.

70.2.3.2 Number of Contractor(s)

To continuously deliver working software in a fast-paced manner, an effective experienced agile team usually develops its own unique internal culture and compatible development tempo. Multiple teams with multiple styles and policies brought in discordant and possibly inharmonious battle rhythm. Potential impediments on a project with several contractors/subcontractors include (a) diseconomies of scale resulting from greater communication challenges with a higher number of team interfaces that need to be managed, leading to a higher likelihood of software integration and team synchronization problems, and (b) possible conflicts among internal development processes and nomenclature across the individual team processes and/or supporting infrastructure that contributes to miscommunication of information and reduced knowledge sharing. Five classifications of number of contractors are:

- Level 1 – Only one contractor using agile software development approach; no subcontractors.
- Level 2 – There are no more than two contractors/subcontractors with close collaboration. Every contractor is using agile software development approach.
- Level 3 – There are no more than four contractors/subcontractors.
- Level 4 – There are no more than six contractors/subcontractors.
- Level 5 – There are more than seven contractors/subcontractors.

70.2.3.3 Project Team Size

To be productive daily throughout the project, scrum master suggests that the development team size should be between three and nine. The possible ramifications of a large agile team are the diseconomies of scale due to lower constructive interaction among a greater number of team members and difficulties in team coordination and knowledge sharing. Five classifications of project team size are:

- Level 1 – One development team; 3 to 9 developers.
- Level 2 – One development team or more than 9 and less than 30 developers.
- Level 3 – More than 1 development team or a total of 30 to 70 developers.
- Level 4 – More than 1 development team; a total of 70 to 250 developers.
- Level 5 – More than 1 development team; a total of more than 250 developers.

70.2.3.4 Supporting Infrastructure and Environment

The agile approach prefers face-to-face conversation [5]. If the development team is not colocated throughout the project, the team should consider whether there is any supporting infrastructure for team collaboration, which includes development and project management infrastructure or sufficient resources for periodical face-to-face meetings. The possible ramifications of a project without team colocation or supporting collaborative infrastructure are (a) delayed feedback due to slow communication loop, (b) diseconomies of scale for synchronization of tacit knowledge leading to reduced knowledge sharing, and(c) higher number of communication challenges in high security domains if the teams are distributed across multiple time zones and must rely on availability of secure facilities and network. Five classifications of supporting infrastructure and environment are:

- Level 1 – All contractor team members are colocated. Collaboration tools and management tools are readily available to support the development. Most interactions are conducted face-to-face.
- Level 2 – Most of the teams are colocated; collaboration tools and management tools are available to support the team. Available face-to-face meetings and telepresence conferencing.
- Level 3 – Some of the teams are colocated; collaboration tools and management tools are available to support the team; limited resources to support face-to-face meetings; most interactions are conducted by telephone. Simple video conferencing is available.
- Level 4 – No colocated teams; collaboration tools and management tools are available to support the team. Limited resources to support face-to-face meetings; most interactions are by telephone.
- Level 5 – No colocated teams; incompatible collaboration and management tools. No resources to support face-to-face meetings.

70.2.3.5 Team Composition

The ideal situation for an agile team is the team that has successfully worked together on previous agile projects. Personnel turnover or a team that has never worked together creates an additional learning curve. The possible ramifications of a project with an inexperienced team or high personnel turnover include (a) unpredictable team velocity, (b) productivity that remains low or unpredictable as significant portions of the development team remain on the steep agile learning curve, (c) inability to build and maintain the team's tacit knowledge, and (d) high context-switching costs and resource availability issues if the developers and development/test resources are multiplexed across programs/projects. Five classifications of team composition are:

- Level 1 – All of the development team members have successfully worked together using an agile approach.
- Level 2 – Some of the development team members have successfully worked together using an agile approach. The development team consists of experienced and adaptable staff with broad capabilities.
- Level 3 – All of the development team members have worked together using a non-agile approach, and they are open to an agile process.
- Level 4 – Some of the development team members have worked together using a non-agile approach, or the development team members are skeptical about an agile process.
- Level 5 – None of the development team members have worked together, or the development team consists of inexperienced personnel or primarily consists of specialists who cannot adapt to alternative work or process.

70.2.3.6 Use of Automated Testing

To be able to deliver working software frequently and sustainably, an agile project requires the use of automated testing. Without automated testing, the project is unable to keep up with the continuous delivery of the product and sustainable development resulting in loss of user feedback early and often creates higher risk of user dissatisfaction with the end product. Five classifications of use of automated testing are:

- Level 1 – Full regression testing and automated testing at every sprint/iteration
- Level 2 – Partial regression testing and automated testing at every sprint/iteration
- Level 3 – Periodic and partial regression testing and some automated testing
- Level 4 – Periodic regression testing during the development
- Level 5 – Regression testing toward the end of the development phase

70.3 Agile Fit Check Assessment

To assess the fitness of a proposal or a project for an agile development, the government team should gather the program status and evaluate the level of conformance to the agile principles by using the 19 factors from the three major categories addressed in Sect. 70.2 and summarized in Table 70.1.

For each factor, select the appropriate level (1–5), as discussed in Sect. 70.2, based on the program status. Use the Agile Fit Check tool to plot the radar graphs. Figure 70.1 shows an example of a dummy project Agile Fit Check assessment result. Each radar chart represents each Agile Fit Check category. The red line represents the program status. The plots that are closer to the center indicate the lower risk or smaller potential impediments of using an agile approach for that

Table 70.1 Summary of Agile Fit Check factors

System's characteristics	Government's level commitment	Contractor's level commitment
1. System criticality	1. Leadership support	1. Developer expertise
2. Requirements volatility	2. Contracting strategy	2. No. of contractor(s)
3. Requirements formality and detail	3. Government expertise	3. Project team size
4. Requirements decomposition	4. Level of oversight	4. Supporting infrastructure and environment
5. Deployment timelines	5. Collaboration	5. Team composition
6. System integration interval	6. End user involvement	6. Use of automated testing
7. Program scope		

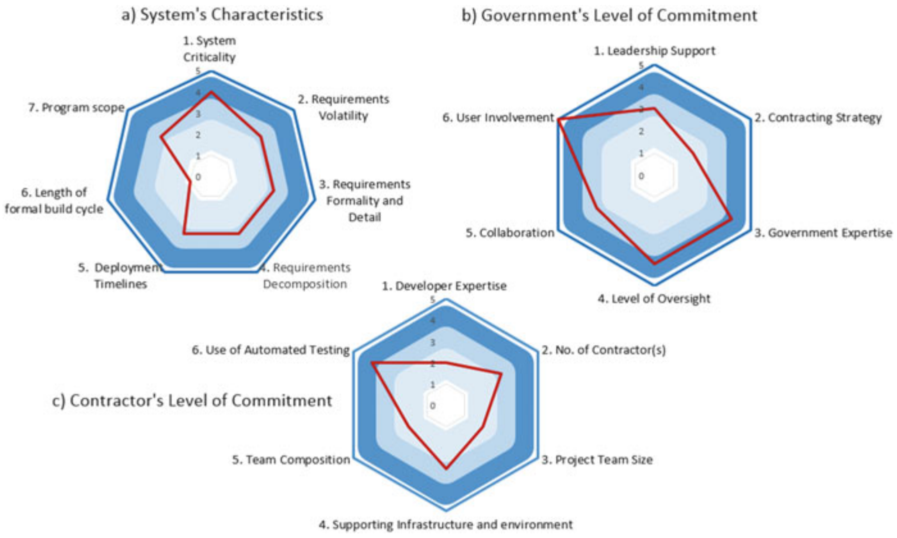


Fig. 70.1 A sample of Agile Fit Check result for (**a**) system's characteristics, (**b**) government's level of commitment, and (**c**) contractor's level of commitment

particular factor. The further from the center, the higher the risk or number of impediments an agile approach would pose.

There is no threshold for agile adoption consideration and this tool does not decide whether the program should or should not follow an agile approach. The next step is to identify high-risk items or the plots that are further from the center. As shown in Fig. 70.1, there are high-risk items that the team should consider and identify potential improvements:

- High system criticality – Although it has been shown that agile can be used in mission/life-critical projects to assure the quality of the system, the government team might need a stronger and closer but a lean oversight and verification and validation process.

- Insufficient agile knowledge on the government team – At the minimum, it is essential for the government team to understand the roles, responsibilities, and expectations of an agile project. Agile trainings and live-in agile coach are popular choices to help the government team to speed up with the learning curve.
- Traditional oversight process – The government team should prioritize the expected deliverables. Traditional metrics, deliverables, and collaboration strategies that delay the process or do not add value to the project should be removed. The government team should also be equipped with appropriate authority, time, and supporting collaborative infrastructure and tools to make quick decision or to provide quick turnaround feedback to the contractor.
- Target users are not available to provide feedback – The government team should identify committed end users or representatives that can be regularly available to provide feedback.
- Insufficient use of automated testing – Automated testing is a mandatory ingredient for agile development. Although not all testing should be done automatically, the agile team should use automated testing where applicable.

Lastly, the government team should discuss and identify possible risk mitigation steps or process tailoring and improvement options. The assessment can be re-run at any time to ensure that the mitigation strategies are progressing in the appropriate direction.

70.4 Conclusion

Agile methods are often viewed as an efficient way to produce deliverables that satisfy customer expectations with a faster schedule. Common challenges with agile development are lack of consistent stakeholder involvement, keeping up with constant fast pace of delivering working software, trust between the government team and the development team, lack of supporting acquisition process, and conflicts between agility, policies, and regulations. Common benefits of agile development are providing early feedback and rapid deliverables, focusing on lean and value-based software development activities and artifacts, higher collaboration and knowledge sharing, and supporting potential changes even late in the game. Key success-critical criteria for agile require ongoing collaboration between all success-critical stakeholders; government and contractor agile knowledge and experience, balancing between flexibility and rigor; frequent and effective discussion and informal reviews; defining success criteria upfront; trust between stakeholders; supporting infrastructure and resources; and continuous improvement.

Not all projects are fit for an agile approach. To evaluate whether a project should adopt an agile approach, the project should be evaluated against the agile manifesto and principles. The use of the Agile Fit Check Framework can help identify the risks of using an agile development method on specific projects. There are three main categories of risk that should be assessed: program or system's characteristics, government's level of commitment, and offeror or contractor's level of commitment. The final step in the Agile Fit Check process is to assist the development team in understanding the potential risks of implementing an agile development method and identifying potential risk mitigation steps and associated process modifications.

References

1. Boehm B, Turner R (2004) Balancing agility and discipline: a guide for the perplexed. Addison Wesley, Boston
2. Modigliani P and Chang S, MITRE defense agile acquisition guide: Tailoring DoD IT Acquisition Program Structures and Processes to Rapidly Deliver Capabilities. MITRE Corporation. March 2014.
3. Manifesto for agile software development. http://agilemanifesto.org/. Accessed 15 Sept 2016
4. Principles behind the Agile manifesto. http://agilemanifesto.org/principles.html. Accessed 15 Sept 2016
5. Cohn M. Agile design: intentional yet emergent. https://www.mountaingoatsoftware.com/blog/agile-design-intentional-yet-emergent. Accessed 15 Sept 2016

Chapter 71
The Agile Systems Framework: Enterprise Content Management Case

James Lockett, Michael Swan, and Kenan Unal

Abstract In this paper, the researchers propose an agile systems framework (ASF) method to obtain quantitative measurements of a system's inherent agility. Agility is defined as the ability of a system to quickly adapt and respond to change. The measurement of agility for information technology systems can help enterprises future-proof their systems. The researchers demonstrate the feasibility of the ASF by showing that interoperability, the ability of two systems to interact effectively (Beesemyer, Empirically characterizing evolvability and changeability in engineering systems, 2012), is positively related to agility. The design of the experiment to test this hypothesis is in an enterprise content management (ECM) case, but the ASF can be applied to different types of systems. The experiment consists of two tests on an ECM system, with the independent variable being the level of interoperability (one test is low interoperability and the other is high interoperability) and the dependent variable being the level of agility measured with the ASF. The experiment provides support for the hypothesis, with the level of agility increasing as the level of interoperability is increased, and shows the feasibility of the Framework. A survey was used to gather expert opinion on the importance of certain criteria of agility used in the ASF, so that the framework accurately measures agility. Further research can test the relationship of agility to other aspects of a system and include more criteria of agility.

Keywords Agile systems framework • Enterprise content management system • Ilities • Interoperability

71.1 Introduction

Systems often lack agility, the ability to respond to future requirements. Lee and Xia wrote in MIS Quarterly that only 11% of Information Systems Organizations were able to keep up with business demands and that 76% were not able to effectively cope with changing business needs, and that this lack of agility can

J. Lockett (✉) • M. Swan • K. Unal
The MITRE Corporation, McLean, VA, USA
e-mail: jdlockett@mtire.org; swan@mitre.org

A.M. Madni et al. (eds.), *Disciplinary Convergence in Systems Engineering Research*, https://doi.org/10.1007/978-3-319-62217-0_71

often result in financial loss [2]. Operations and maintenance (OM) costs proliferate as new systems are added to meet changing requirements. In February 2015, the U.S. General Accountability Office (GAO) reported on the growing, and unsustainable, share of IT budgets being consumed by O&M of existing systems. The GAO found that 27 federal agencies planned to spend almost three quarters of their federal IT budget on the O&M of legacy investments [3]. Designing agility into systems can reduce this cost proliferation.

The agile systems framework (ASF) can be used to assess the intrinsic systems agility of implemented and future systems designs. The framework's use for measuring a system's agility can reduce system cost proliferation by enabling the measurement of nonfunctional requirements, or requirements for how a system shall perform an action [4]. This paper's research suggests that the framework can be effective in measuring a system's agility by determining the relationship between agility and interoperability. In this paper, interoperability is identified as a nonfunctional requirement and is defined by Beesemyer as the ability of a system to effectively interact with other systems [1]. The researchers acknowledge there are many different types of nonfunctional requirements. For this research, interoperability was chosen because there is already a credible model in place to measure it. The hypothesis that interoperability is positively correlated with agility is tested. The following section details the literature review that led to the development of the ASF and the hypothesis for which to test the framework's effectiveness. Section 71.3 outlines the method and experiment procedure to test the hypothesis. Section 71.4 shows the results of the experiment. The final section of the paper presents a discussion of conclusions drawn from the experiment.

71.2 Literature Review

The definition of "agility" this paper uses pulls from the work of Habernfellner and de Weck and how they define agility. Their systems engineering approach defined agility in the context of describing both a system engineering process and the agility of a system itself [5]. Following the publication of "The Agile Manifesto," agility became a popular term in software development [6, 7]. From the enterprise point of view, agility represents a business ability to compete, particularly through employing information technology (IT) [8, 9]. Much of the systems engineering research on agility focuses on enterprise, software, and systems development approaches. What is addressed in this paper is the less studied agility intrinsic to system architectural implementations and draws on early agile enterprise research for a top-level conceptual construct.

In 1991, Nagel and Dove led a U.S. government-funded cross-industry study focused on the agile manufacturing enterprise, where the term "agile enterprise" was coined [10, 11]. The authors defined the agile enterprise as an organization that has the flexibility to adopt for each project the managerial vehicle that will yield the greatest advantage. They also stated that in an agile enterprise, machinery can be reprogrammed quickly to produce new products in many variations [10]. Networks, information, and communications systems were key parts of this focus on the agile manufacturing enterprise. Dove expanded the definition of the agile enterprise to be an organization, which must manage and apply knowledge. This effectively balances the intellectual ability to find appropriate things to act on, or knowledge management, and the ability to dynamically and competently act on a change, or change proficiency, in business systems and processes as necessary elements of agility [12]. Change proficiency evolved into response proficiency as a construct to represent the ability to respond effectively to an uncertain and unpredictable environment [13].

Dove and LaBarge decomposed the response proficiency of a system into two dimensions: proactive and reactive proficiency. They defined proactive proficiency as the degree to which a system is composable from system elements (e.g., building blocks) and reactive proficiency as the degree to which a system is resilient in response to an environmental change [9]. They related agility to composability with Silletto's work on military "composable capability" where Stilleto emphasized taking a "system focus" rather than looking at individual elements when considering composable capability [14]. Dove and LaBarge also related agility to resilience with Albert's study of agile command and control, stating that resilience is one of six components of agile systems and gives the ability to repair, replace, patch, or otherwise reconstitute lost performance [15].

Based on Dove and LaBarge's study, Fig. 71.1 shows the degree to which a system is agile in terms of composability and resilience.

Boehm extended agile systems research further in a study conducted for the Systems Engineering Research Center (SERC) ilities Tradespace and Affordability – Phase 1 (iTAP). In the iTAP study, ilities are defined as hierarchies of system-wide properties that specify how well a system should perform [16]. Ross and Rhodes extended the iTAP study and placed composability and resilience in a "Top Level List" set of ilities, shown in Fig. 71.2 [17].

The variety in ility definitions and relationships is a necessary consideration when decomposing response proficiency, composability, and resilience into ilities. The iTAP study reported a plethora of inconsistent and often inadequate definitions of ilities [16]. Ross and Rhodes point out that gaps in shared understanding of ilities can lead to costly project and operational failures [18]. Recent research has led to

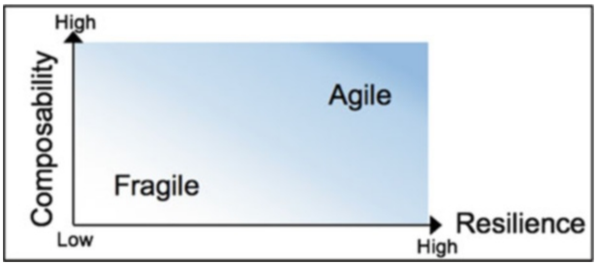


Fig. 71.1 Two dimensions of response proficiency [10]

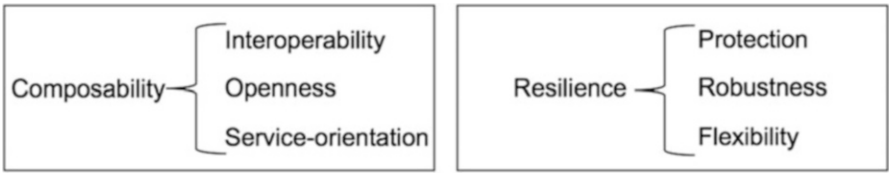


Fig. 71.2 Ilities mapping to composability and resilience [17]

Table 71.1 Change-related and architecture-related ility examples [2]

Change-related ilities	Architecture-related ilities
Adaptability	Accessibility
Agility	Controllability
Changeability	Decentralization
Evolvability	Independence
Extensibility	Interoperability
Flexibility	Integrality
Modifiability	Modularity
Re-configurability	Protectability
Scalability	Readability
Survivability	Redundancy
Versatility	Simplicity

more consistent and scientific models for describing and understanding ilities and their role in system change [18–22]. The iTAP study also built on this research [17].

The SERC-associated research at the Massachusetts Institute for Technology (MIT) argues that there are a least three types of ilities: change-related, architecture-related, and new ability-related [21]. Beesemyer organized change-related and architecture-related ilities, which are provided in Table 71.1.

Ross and Rhodes developed a working hypothesis that architecture-related ilities are enablers for change-related ilities from earlier work on how and why ilities are related to one another. In their work, a quantitative review of citation frequency and preliminary elicitation of means-ends hierarchical relationships among ilities show semantic sets of ilities. Ross et al. identified interoperability as an architecture-related ility and as part of a set supporting the change-related ility composability [17]. Ross and Rhodes defined composability as a “Top Level ility.” Boehm also categorized interoperability under composability [16]. Dove and LaBarge’s work shows the relationship between agility and composability [9, 17]. The research, development, and automation of ility models, prescriptive methods, and ontology are ongoing efforts that cause the evolution of ility definitions and theories.

The researchers follow the iTAP/MIT ility research definitions and ility hierarchies to identify and measure intrinsic systems agility. Intrinsic systems agility measurements based on the above recent semantics and definitions can provide an independent measure of the efficacy of the ility research approaches affecting structural architecture-related ilities. The researchers also present an approach for the measurement of interoperability and then assess the relationship between agility and interoperability. Basing this research on the study of ilities by Ross and Rhodes lays a foundation for testing their working hypothesis that architecture-related ilities (e.g., interoperability) are enablers for change-related ilities (e.g., agility) [17].

71.3 Methodology

This paper tests the following research hypothesis:

- Hypothesis: Systems interoperability has a positive correlation to systems agility.
- Null hypothesis: No relation exists between systems interoperability and systems agility.

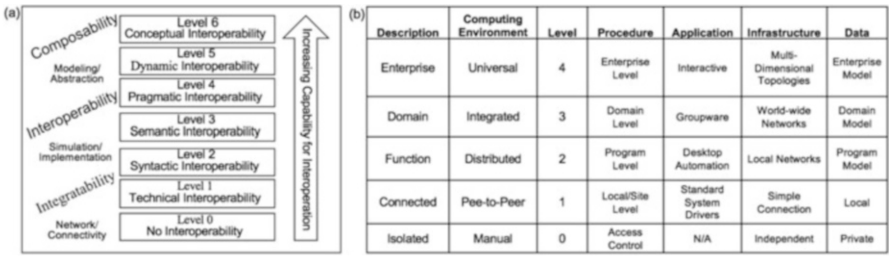
The researchers answered three questions to test the hypothesis:

- How can systems architecture be measured to show a level of interoperability?
- Which attributes are the most important contributors to systems agility?
- What is the relationship between systems interoperability and systems agility?

71.3.1 *How Can Systems Architecture Be Measured to Show a Level of Interoperability?*

The researchers use the levels of conceptual interoperability model, or LCIM, as shown in Fig. 71.3, to answer this question. The LCIM is a reference model, which organizes systems interoperability into seven layers to better understand interoperation between systems [23].

However, the LCIM does not specify any characteristics within a layer to measure the level of interoperability. A mature interoperability model, the levels of information system interoperability (LISI) model provides an example of a


(a)
Composability
Modeling/ Abstraction
Interoperability
Simulation/ Implementation
Integratability
Network/ Connectivity
Level 6 Conceptual Interoperability
Level 5 Dynamic Interoperability
Level 4 Pragmatic Interoperability
Level 3 Semantic Interoperability
Level 2 Syntactic Interoperability
Level 1 Technical Interoperability
Level 0 No Interoperability
Increasing Capability for Interoperation
(b)

Description	Computing Environment	Level	Procedure	Application	Infrastructure	Data
Enterprise	Universal	4	Enterprise Level	Interactive	Multi-Dimensional Topologies	Enterprise Model
Domain	Integrated	3	Domain Level	Groupware	World-wide Networks	Domain Model
Function	Distributed	2	Program Level	Desktop Automation	Local Networks	Program Model
Connected	Pee-to-Peer	1	Local/Site Level	Standard System Drivers	Simple Connection	Local
Isolated	Manual	0	Access Control	N/A	Independent	Private

Fig. 71.3 (**a**) Tolk's levels of conceptual interoperability model [23] and (**b**) The levels of information systems interoperability (LISI) model [24]

model with measurable characteristics [24], which can be mapped into the appropriate layers of the LCIM. The LISI model assigns a quantitative level to a system's interoperability based on four capabilities: procedure, application, infrastructure, and data. The LISI model layers – isolated level, connected level, functional level, domain level, and enterprise level – can be mapped to the LCIM layers, effectively incorporating the capabilities used in the LISI model.

For a given system, a qualitative analysis is performed where the examiner, knowledgeable in IT systems, rates a system on its capability with regard to procedure, applications, support infrastructure, and handling data. The rating is on a 5-point scale, ranging from 0 to 4 and divided further into alphabetical levels. A high level of interoperability will have a higher number score and letter further in the alphabet. Within a technical domain, the rating provides an interoperability figure of merit for both standalone and comparison purposes.

71.3.2 Which Attributes Are the Most Important Contributors to Systems Agility?

The agile systems framework (ASF) was created based on the work of Soumia, Rabah, Adbelaziz, and Mohamed, as well as Ross and Rhodes. An advantage of the ASF is that agility can be measured by quantitative metrics and provide credible assessments when analyzing, comparing, and future-proofing systems.

The researchers selected the agility assessment method, a conceptual analysis grid of agility evaluation proposed by Soumia et al. because it can define a unique set of dimensions and associated criteria [25]. The evaluation metrics for agility criteria in the conceptual analysis grid was normalized on a Likert-5 scale (1–5), which the ASF also uses. The basis of the ASF comes from nonfunctional requirements expressed as ilities or criteria, organized into a "Semantic Basis" set. Nonfunctional requirements of a system determine its overall degree of agility [1]. The researchers pulled the definitions of criteria from Ross and Rhodes' Semantic Basis for change-type ilities model, which is based off of Beesemyer's work [1, 17]. The researchers use the flexibility and robustness dimensions as those dimensions are used in Soumia et al.'s analysis grid and are mapped to resilience by Boehm's research, which Dove and LaBarge also related to agility [9, 16]. - Re-configurability does not appear in Soumia et al.'s analysis grid, but the definition for re-configurability in Ross and Rhodes' Semantic Basis matches very closely to the criteria used to measure Integration in Soumia et al.'s analysis grid [17, 25]. Consequently, the researchers chose to use the criteria for integration for re-configurability within the ASF given how closely associated they are to one another. Measuring the overall agility of a system also requires computations from the metrics found with the ASF. The researchers follow Soumia et al. to measure systems agility as follows [25].

The agility of a given dimension (D_i) is measured by the average of the metric of the kth criterion of D_i (C_k) for the total number of criteria of D_i (NC). The ASF determines the criteria as in Eq. (71.1):

$$A_{D_i} = \left(\sum\nolimits_{k=1}^{NC} C_K\right)/NC \tag{71.1}$$

The overall agility of the system (A_S) is the average of the ith dimension of the system for the total number of dimensions in the system (ND) as given in Eq. (71.2):

$$A_S = \left(\sum\nolimits_{i=1}^{ND} {A_{D_i}}^{i}\right)/ND \tag{71.2}$$

To validate the importance of the dimensions and criteria used in the ASF, the researchers conducted a survey. The survey was modeled after an approach conducted by Pitsko [26]. The survey measures responses on a Likert-5 scale (1–5), where 1 is "Not at all Important" and 5 is "Extremely Important." The survey asks the respondents to name an IT system they are very familiar with. Next, the survey asks the respondents to measure the overall importance of flexibility, robustness, and re-configurability with their IT system in mind, when the definitions of the dimensions are given. The definitions of flexibility, robustness, and re-configurability are from Ross and Rhodes' Semantic Basis [17]. The survey also asks respondents to measure the importance of criteria associated with flexibility, robustness, and re-configurability, taken from Soumia et al., with their IT system and given previous definitions [25]. Finally, the survey asks the respondents to rate the importance of change-related ilities defined by Beesemyer [1]. In this question, the responders are only given the definitions of the ilities, rather than the names, so as to not incur any biases. This question measures how well the ility terms, ascribed by ility researchers, properly and clearly define elements of agility. If responders answer this question with a low score, it indicates that these ility definitions do not resonate with users when they consider the importance of systems agility. One final note about the survey is that it does not ask respondents for any personal, identifiable information.

The survey was deployed in July 2016. The goal of the survey was to achieve a confidence level of 95% with a confidence interval of 10. As of January 2017, there have been 70 responses resulting in a confidence interval of 11.71 given an estimated population of 10,000 IT government workers. Survey respondents were asked to classify themselves among different government agencies and by their job role. The leading government agency identified was the "Department of Defense." The leading role respondents identified with was "engineer."

Given the small sample size of the survey, only basic exploratory analysis could be conducted on the data. In order to determine if the three dimensions of agility used in the ASF (flexibility, robustness, and re-configurability) are internally consistent and maintain proper reliability, Bivariate correlation analysis was used to determine if the criteria for each dimension are reliable in measuring some

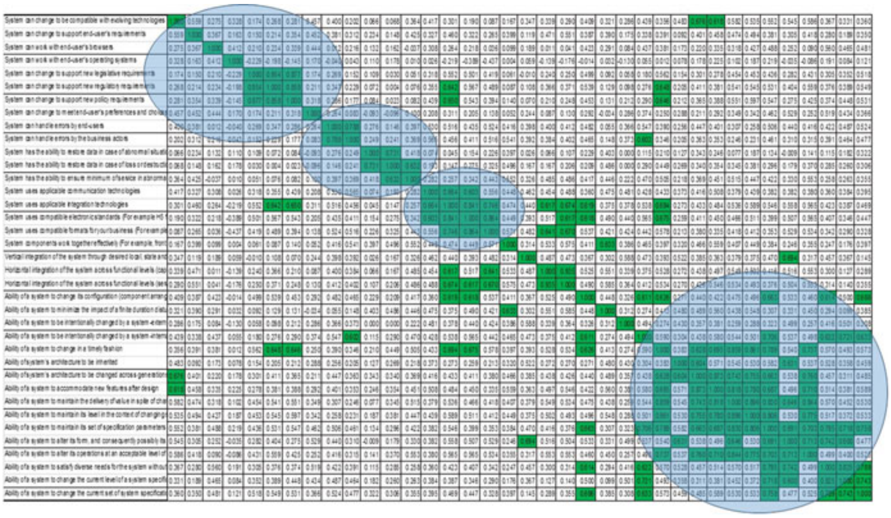


Fig. 71.4 Correlations across all criteria and dimensions

consistent underlying construct. Because the sample size was so small, it is not possible to perform any more robust statistical tests than these basic exploratory tests.

The first step in assessing the data to determine underlying associations is to use simple correlations to determine how strongly associated the criteria are with the dimensions and ility definitions in the proposed index. Figure 71.4 shows the correlations across all the criteria, dimensions, and definitions asked in the survey.

The four shaded regions, from left to right, are flexibility, robustness, re-configurability, and the ilities definitions by Beesemyer [1]. In this case, the questions identified with robustness and re-configurability are all correlated at the $p = 0.05$ level, with one exception ("the ability to restore data in case of abnormal situations" and "can handle errors by end-users" are correlated at $p = 0.051$). Questions regarding flexibility, although somewhat correlated, do not show the same strength of association, and one question ("system can change to support new legislative requirements") showing no correlations to other elements of the index.

The correlation coefficients of those indices were computed for each ility defined by Beesemyer to determine which of Beesemyer's ilities are associated with the categories constructed for the research. At the $p = 0.05$ level, all but two of the Beesemyer ilities were correlated with all three of the constructed indices. Of those two, survivability was correlated with both robustness and re-configurability, and (Beesemyer's) flexibility was correlated with re-configurability (and notable, not with the constructed flexibility index) .

71.3.3 What Is the Relationship Between Systems Interoperability and Systems Agility?

The researchers apply the LISI model and ASF in an experiment where the independent variable is the interoperability level of the system and the dependent variable is the agility level of the system. The LISI model and ASF can quantify the interoperability and agility levels in the following steps. The researchers start with measuring the interoperability of an IT system with the LISI model and define it as IOP_o. The researchers then measure the starting level of agility with ASF and denote it as $A_{S,o}$. Next, the researchers adjust the environment with increased interoperability and measure the new level of interoperability again with the LISI model. The researchers define this new level as IOP_n. Then the researchers measure the system's new agility level with the ASF, denoted as $A_{S,n}$. Finally, the researchers compare the relationship between IOP_o and $A_{S,o}$ to IOP_n and $A_{S,n}$ to verify the hypothesis that interoperability has a positive correlation to agility. The experiment procedure is shown in Fig. 71.5.

The researchers use the enterprise content management systems (ECMSs) Drupal and Alfresco in the experiment. Drupal is an IT system that is used to create websites, sharing gathered information to users over the Internet. Alfresco is an IT system that can be used as a repository of information; storing and editing it by one or many users over the Internet. Both ECMSs can work together, with Alfresco acting as a repository of information shared by a group of users and Drupal acting as a space to share that information to the public through a website. Many organizations use both ECMS and since they are open source, they are easy to manipulate. Hence, the results of the experiment are relevant to many organizations.

There are two tests in the experiment. The first test involves transfer of data files from an Alfresco repository to a Drupal front-facing website in their base forms. In the second test, data files are transferred again, but with the content management interoperability services (CMIS), application downloaded. CMIS allows ECMSs like Drupal and Alfresco to be integrated, giving the two systems file sharing abilities, which is a good indicator of increased interoperability. The tests performed are shown in Fig. 71.6.

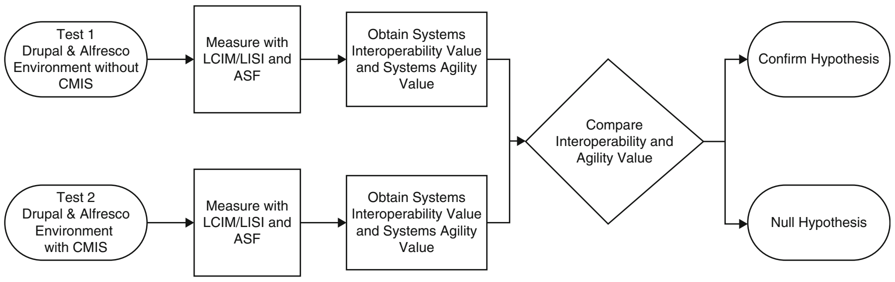

Test 1
Drupal & Alfresco Environment without CMIS
Measure with LCIM/LISI and ASF
Obtain Systems Interoperability Value and Systems Agility Value
Test 2
Drupal & Alfresco Environment with CMIS
Measure with LCIM/LISI and ASF
Obtain Systems Interoperability Value and Systems Agility Value
Compare Interoperability and Agility Value
Confirm Hypothesis
Null Hypothesis

Fig. 71.5 Experiment procedure diagram

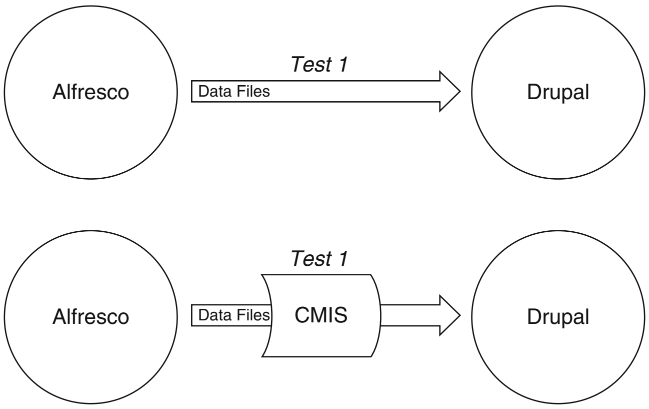


Fig. 71.6 Experiment test process diagram

To measure interoperability with the LISI model, a team of engineers, experienced in IT systems, analyzes the system on its abilities in procedure, application, infrastructure, and data and assigns a level to each of the four capabilities. The highest level achieved by all four columns is the overall level of interoperability for the system. To measure agility with the ASF, the same team of engineers assigns a score to each criterion on the grid based on how well the system achieves that criterion. When all the scores are assigned, use Eq. (71.1) to calculate AD for each of the three dimensions (flexibility, robustness, and re-configurability). Then take the average of the criteria scores for each of the dimensions, and calculate AS using Eq. (71.2).

Note that the hypothesis stated by this paper can relate to the hypothesis developed by Ross and Rhodes in that both focus on the interaction between interoperability and agility. The results of the experiment to test the paper's hypothesis, therefore, is a starting point in testing Ross and Rhodes' hypothesis that architecture-related ilities are enablers of change-related ilities [17].

71.4 Results (Table 71.2)

Table 71.3 shows the results of Tests 1 and 2. Appendix A shows the calculations.

In the ASF, there are three dimensions to systems agility: flexibility, robustness, and re-configurability. The agility score for robustness did not change from Test 1 to Test 2; it was rated both times as 2. The agility score for flexibility changed slightly between the two Tests by a quarter of a point (3 for Test 1 and 3.25 for Test 2). It was re-configurability that had a significant change, with an agility score of 2.375 for Test 1 and 4 for Test 2.

During Test 1, the file transfer from Alfresco to Drupal was done by downloading the files from the Alfresco repository, then copying them into the

Table 71.2 Results of experiment

Experiment tests	Systems interoperability value (*IOP*)	Systems agility value (A_S)
Test 1	2c	2.458
Test 2	3b	3.083

Table 71.3 ASF analysis grid completed for Tests 1 and 2

Dimensions	Criteria	Test 1	Test 2
	System can change to be compatible with evolving technologies	3	4
	System can change to support end-user's requirements	4	4
	System can work with end-user's browsers	5	5
Flexibility	System can work with end-user's operating systems	4	4
	System can change to support new legislative requirements	0	0
	System can change to support new regulatory requirements	0	0
	System can change to support new policy requirements	4	5
	System can change to meet end-user's preferences and choices	4	4
Agility of dimension		*3*	*3.25*
	System can handle errors by end-users	2	2
	System can handle errors by the business actors	2	2
Robustness	System has the ability to restore data in case of abnormal situations	3	3
	System has the ability to restore data in case of loss or destruction	2	2
	System has the ability to ensure minimum of service in abnormal situations	1	1
Agility of dimension		*2*	*2*
	System uses applicable communication technologies	5	5
	System uses applicable integration technologies	2	5
	System uses compatible electronic standards	2	4
Re-configurability	System uses compatible formats for your business	5	5
	System components work together effectively	1	5
	Vertical integration of the system through desired local, state, and federal systems	0	0
	Horizontal integration of the system across functional levels (capabilities) with other systems	2	4
	Horizontal integration of the system across functional levels (services) with other systems	2	4
Agility of dimension		*2.375*	*4*
Total degree of agility of the system		*2.458*	*3.083*

Drupal site. This was a time-consuming and inefficient process. During Test 2, CMIS allowed the Drupal site to automatically copy and transfer files from Alfresco with the only manual input being to transfer the files to Alfresco first.

CMIS allows file sharing between Drupal and Alfresco, so it is reasonable that flexibility and re-configurability would be greater when the application is installed. But file sharing is the only ability the application gives, so robustness would stay the same even with the application installed.

71.5 Conclusions

There has been relatively little research examining the measurement of agility in IT systems, even though doing so can help an organization future-proof its systems and reduce O&M costs. The researchers proposed a new approach, the agile systems framework (ASF), to measure the agility of an IT system and assessed the effectiveness of this approach. The effectiveness of the ASF was assessed by testing the hypothesis that interoperability is positively correlated to agility. The researchers posed three questions that must be answered to test the hypothesis. The researchers answered these questions using the existing LISI model to measure interoperability and created the ASF to measure agility. A survey was conducted to assess the feasibility of using the ASF to measure agility. Preliminary survey results indicated strong correlation with the robustness and re-configurability dimensions of agility. However, further refinement is needed with the flexibility criteria in the ASF.

An experiment was performed using these models to test the hypothesis in an IT environment using ECMSs. The results of the experiment show a positive relationship between systems interoperability and systems agility. With the ASF, there is now the beginning of a functional framework to measure systems agility to meet future requirements. This research also supports Ross and Rhodes' hypothesis that architecture-related ilities are enablers of change-related ilities since the results here demonstrate an instance of an architecture-related ility, interoperability, being an enabler for change-related ility, agility.

Future work should include different experiments designed to measure interoperability and agility, as well as using different ilities to study their relationship with systems agility. The researchers expect to obtain more responses to the existing survey to achieve a confidence interval of 10. Enhancements to the survey can be made to incorporate modified definitions to Beesemyer's change-related ilities. Additional surveys could be conducted to perform a cross-longitudinal study beyond users of government IT systems.

Acknowledgments This research was supported by Dr. Chuck Watson, who provided research review and critiquing, and Dr. John Bornmann, who provided statistical analysis of the survey results.

Appendix A: Experiment Calculations with the ASF

Below are ASF assessments for Test 1 of the Drupal and Alfresco system without CMIS installed and Test 2 of the Drupal and Alfresco system with CMIS installed.

References

1. Beesemyer Jr JC (2012) Empirically characterizing evolvability and changeability in engineering systems. Massachusetts Institute of Technology
2. Lee G, Xia W (2010) Toward agile: an integrated analysis of quantitative and qualitative field data on software development agility. MIS Q 34(1):87–114
3. Government Accounting Office (2015) High-risk series, an update, report GAO-15-290
4. Eckhardt J, Vogelsang A, Fernández DM Are non-functional requirements really non-functional?: an investigation of non-functional requirements in practice. Paper presented at: 38th International Conference on Software Engineering; Austin, TX, 14–22 May 2016, p 832–842
5. Haberfellner R, Weck O (2005) 10.1. 3 Agile systems engineering versus agile systems engineering. INCOSE International Symposium 15(1):1449–1465
6. Highsmith J, Cockburn A (2001) Agile software development: the business of innovation. Computer 34(9):120–127
7. Chuang SW, Luor T, Lu HP (2014) Assessment of institutions, scholars, and contributions on agile software development (2001–2012). J Syst Softw 93:84–101
8. Sambamurthy V, Bharadwaj A, Grover V (2003) Shaping agility through digital options: reconceptualizing the role of information technology in contemporary firms. MIS Q 27 (2):237–263
9. Dove R, LaBarge R Fundamentals of agile systems engineering-part 1. Paper presented at: International Council on systems engineering, International Symposium; Las Vegas, NV, Jun 30 – Jul 31 2014
10. Nagel RN, Dove R (1991) 21st century manufacturing enterprise strategy: an industry-led view, 1st edn. Diane Publishing, Bethlehem
11. Imache R, Izza S, Ahmed-Nacer M (2012) An enterprise information system agility assessment model. ComSIS 9(1):108–133
12. Dove R (1999) Knowledge management, response ability, and the agile enterprise. J Knowl Manag 3(1):18–35
13. Dove R (2001) Response ability: the language, structure, and culture of the agile enterprise, 1st edn. John Wiley & Sons, New York
14. Sillito HG Composable capability – principles, strategies and methods for capability systems engineering. Paper presented at: International Council on systems engineering, International Symposium; Philadelphia, PA, 24–27 Jun 2013
15. Alberts DS (2011) The agility advantage: a survival guide for complex enterprises and endeavors. DoD Command and Control Research Program (CCRP). p 217–218
16. Boehm B (2013).Tradespace and affordability-phase 1 (No. SERC-2013-TR-039-2). Systems Engineering Research Center, Hoboken
17. Ross AM (2014) Contributing toward a prescriptive theory of ilities. In 1st Annual SERC Technical Review. RT-113 Foundations
18. Ross AM, Rhodes DH (2015) Towards a prescriptive semantic basis for change-type ilities. Procedia Comput Sci 44:443–453
19. Fricke E, Schulz AP (2005) Design for changeability (DfC): principles to enable changes in systems throughout their entire lifecycle. Syst Eng 8(4):S.342–S.359

20. Ross AM (2006) Managing unarticulated value: changeability in multi-attribute tradespace exploration. Massachusetts Institute of Technology, p 361
21. Ross AM, Rhodes DH, Hastings DE (2008) Defining changeability: reconciling flexibility, adaptability, scalability, modifiability, and robustness for maintaining system lifecycle value. Syst Eng 11(3):246–262
22. De Weck OL, Ross AM, Rhodes DH Investigating relationships and semantic sets amongst system lifecycle properties (ilities). Paper presented at: 3rd International Engineering Systems Symposium; Delft, Netherlands, 18–20 Jun 2012
23. Tolk A, Bair LJ, Diallo SY (2012) Supporting network enabled capability by extending the levels of conceptual interoperability model to an interoperability maturity model. The Journal of Defense Modeling and Simulation: Applications, Methodology, Technology 10(2):145–160
24. C4ISR Architecture Working Group (1998) Levels of information systems interoperability (LISI). OSD(ASD(C3I)). Washington, DC
25. Soumia A, Rabah I, Adbelaziz K, Mohamed M (2013) A components oriented method for evaluation of e-government information systems agility. The Research Journal of Recent Sciences 2(7):56–65
26. Pitsko R (2013) Principles for architecting inherently adaptable complex systems. Stevens Institute of Technology, p 263–280

Chapter 72
Quantifying the Ilities: A Literature Review of Robustness, Interoperability, and Agility

Andrew J. Turner, William Monahan, and Matt Cotter

Abstract This paper presents a literature review of the methods for quantifying system ilities. This paper is the second in a series that quantifies the ilities. The ilities are nonfunctional system characteristics. The motivation for designing systems to express these ilities is in response to the ever increasing complexity of engineered systems, increasing requirements on budget and schedule, and the need to adapt to a rapidly changing world. The systems-engineering community has increased its focus on defining and implementing the ilities for over a decade. Many works available in the literature have defined the ilities and their inter-relations. A subset has gone further to quantify the ilities; however, the literature is currently fragmented in its approach. This paper attempts to identify and summarize the various approaches in quantifying the ilities. A systematic literature review was conducted across eight conferences and journals from 2010 to 2015. This paper addresses the findings for robustness, interoperability, and agility, which include definitions or quantifications in 222 documents. The definitions discovered for each of the ilities demonstrated relative agreement; however, many differed in the details. Robustness was found to have a wide number and variety of quantifications. A moderate number of quantifications were found for interoperability, whereas few quantifications were found for agility. It is recommended that further efforts by the community are devoted to the development of quantified measures for interoperability and agility.

Keywords Ilities • Robustness • Interoperability • Agility • Quantification • Definition

72.1 Introduction

The term ilities refers to a class of properties that are used to describe nonfunctional system characteristics [1]. The need for improving the ilities of systems has been called for by system operators, technical leaders, and political leaders [2–6].

A.J. Turner (✉) • W. Monahan • M. Cotter
The MITRE Cooperation, M/S170, 202 Burlington Road, Bedford, MA 01730, USA
e-mail: ajturner@MITRE.org

A.M. Madni et al. (eds.), *Disciplinary Convergence in Systems Engineering Research*, https://doi.org/10.1007/978-3-319-62217-0_72

The motivation for designing systems with the ilities in mind is many fold. Much of the need arises in response to the ever increasing complexity of engineered systems, increasing requirements on budget and schedule, and the need to adapt to a rapidly changing world [7, 8]. However, delivering these capabilities has proven to be difficult. One such difficulty arises from the fact that the meaning of the various ilities differs based on the party using the term, and often the terms are used without explicit definition [4]. Many works are available in the literature that address this issue and provide a taxonomy of definitions for the ilities [1, 4, 9]; however, quantitative measures of the ilities are uncommon. The literature is currently fragmented in its approach to quantifying the ilities. Therefore, this paper attempts to identify and summarize the various approaches to quantify the ilities found in the systems-engineering literature. The paper reports findings on the following ilities: robustness, interoperability, and agility. This paper is the second in a series that addresses quantification of the ilities. The first covered resiliency, flexibility, and adaptability and was presented at CSER 2016 [10]. The remainder of this paper is as follows. Section 72.2 covers the process used for the literature review. Sections 72.3 through 72.6 provide the summaries of the findings for robustness, interoperability, and agility, respectively. These sections are divided into subsections on definitions and quantifications. Finally, the paper concludes with a discussion on the state of the quantification of the ilities.

72.2 Literature Review Method

A systematic review of the literature was conducted across eight conferences and journals from 2010 to 2015. The text of each article was searched for the terms: robustness, robust, interoperability, interoperable, agility, and agile. The resulting set of articles was manually scanned for definitions or quantifications. The review uncovered 150 articles that either define or quantify robustness, interoperability, or agility. The remaining articles were then reviewed to determine how the ilities were defined and quantified.

The eight conferences and journals searched were Complex Adaptive Systems Conference, Conference on Systems Engineering Research, INCOSE International Symposium, International Journal of System of Systems Engineering, Systems Conference, Systems Engineering Journal, IEEE Systems Journal, and System of System Engineering Conference. There are several omissions from the reviewed dataset due to conference or journal inexistence at the time of data collection, lack of an electronic database, or existence of a paywall. These omissions are Complex Adaptive Systems Conference 2010 and 2015, Conference on Systems Engineering Research 2011, Systems Conference 2010, INCOSE International Symposium 2015, and the International Journal of System of Systems Engineering 2010 and 2011.

72.3 Robustness

72.3.1 Robustness Definition

The literature review uncovered 69 articles that either define or quantify robustness. The literature generally agrees on the definition of robustness. Robustness can be categorized into two application areas: robust systems and robust designs. The first refers to a system's ability to maintain its function or performance level in the face of change after it has been fielded. The second refers to the selection of a system design that either remains the best design across different selection criteria, or the system performance does not vary in the face of known uncontrollable factors prior to being fielded.

A robust system can broadly be defined as the ability of a system to maintain capability in the face of change after it has been fielded. The literature varies in their treatment of maintain capability and change. There are three themes that describe a system's ability to maintain capability. First, the performance of the system is insensitive to change [9, 10, 12]. For example, a system is capable of operating in a variety of environmental conditions, e.g., a bridge is capable of functioning year round. Second, the system continues to function in the face of change [9, 11, 13–15]. For example, a system may have a threshold value, which if violated, will cause the system to cease functioning. If the system does not completely fail in the face of this change, then it may be considered robust, e.g., adding weight to a bridge. Third, the degradation of the system is graceful in the face of change [11]. For example, an overloaded bridge that instantaneously collapses once the weight threshold is reached is not a graceful degradation and may not be considered robust. These three themes are not exclusive. A system's robustness can be assessed using any combination of these themes.

These effects on the system's capabilities are brought about by some change. This change has a source and a form. The source of the change can originate internally or externally to the system. The form of change can be a modification to something with which the system interacts, or a modification of the system itself. Illustrative examples of these types of changes to different systems are shown in Table 72.1.

A system cannot be simply defined as robust, there are various ways to measure a system's ability to maintain capability, and there are numerous types of change the system can be subjected to. Instead, the robustness of a system is multidimensional. The assessment of robustness is meaningless without corresponding information

Table 72.1 Examples of types of change to a system

	Change to a system interaction	Change to the system
Internal source	The power demand in a regional energy grid changes	A server fails in a computer network
External source	Weather changes around an airport	A naval fleet loses a destroyer in a military engagement

into how the system maintains capability and to what changes it is subjected [8, 11]. Given that it is unknown as to what changes a system will be subjected to once it is fielded, a system's robustness is truly known only after the system is fielded [9, 12, 13].

Though robustness of a system is only accurately assessable after it is fielded, a system can still be designed for robustness. Commonly, a system design is considered robust when it is insensitive to known uncontrollable factors [14, 15]. For example, military radios should be designed to be insensitive to changing atmospheric conditions. Alternatively, a robust design may also refer to the design that remains the best choice of a set of designs as assumptions and evaluation metrics are changed [16, 17]. For example, a robust automobile design would appeal to a large number of customers, independent of their selection criteria, e.g., fuel economy or power.

72.3.2 Robustness Quantification

The literature review found a significant number of quantifications for robustness; however, there exists a wide variety of approaches to quantification. The quantifications found in the literature can be loosely categorized into performance measures, design measures, and graph theoretic measures, where performance measures and design measures are related. All quantifications can be applied to the definition of robust designs; performance measures and graph theoretic measures can be applied to robust systems.

Several quantifications of robustness use measures of system performance. The works by Pape et al. measure the robustness of a system of systems (SoS) by the maximum performance loss when any one component system is removed [13, 18]. Acheson et al. quantify robustness as the percent degradation of key performance parameters as a SoS develops and evolves [19]. Similarly, Guariniello and DeLaurentis use the percentage of maximum operability lost when faced with changes in the operability of other systems in the SoS [20] or as systems join or leave the SoS [21]. Gomez et al. use variance of their SoS performance, caused by changes in the number of participating systems [22]. Muller defines robustness as the rate of change of system performance [23]. These quantifications of robustness can be combined to be thought of as a performance distribution over time, where the measures used for robustness are expected value, variance, and rate of change of the system's performance.

Most of the work found concerning design robustness related to the system's insensitivity to changes. This is similar to the definitions of robust performance. These measures of design robustness have been greatly influenced by Taguchi's work; though only a few in our defined review set referenced him directly [11, 15]. Squires and Cloutier use Taguchi's methods in the development of an online systems-engineering course [15]. Though Taguchi's work is outside the scope of the defined review set, a discussion on quantitative robustness measures would be incomplete without it. Taguchi's methods rely heavily on a loss function, e.g., the quadratic loss function. The quadratic loss function defines some target value, t, a design or process is trying to achieve. The actual response, y, is a function of controllable variables, x, and noise variables, z. The equation for minimizing the expected value of the quadratic loss function is defined as: $x^* = \arg\min_x \left(E\left[(f(x,z) - t)^2\right]\right)$ [11]. The expected loss function can then be rewritten as a combination of the mean and variance of the loss function and the target value: $(f(x,z) - t)^2 = \sigma_y^2 + \left(\mu_y - t\right)^2$ [11].

Therefore, a design is more robust when the response value is closer to the target value and the variance of the response is reduced. These are common themes in design robustness. Dickerson and Mavris use a performance margin $(Y_i - y_i)$, defined as the difference between the system's response value, y_i, and the system's requirement, Y_i, for all requirements i [24]. Note that their method is a constraint instead of a target value; however, the concept is the same. Additionally, Gomez quantified the SoS robustness as the increase in variance due to changes in the SoS [25]. Finally, Sitterle et al. use a combination of spearman correlation of the design performance measures against the noise variables and measures of local covariance as an indication of design robustness [12]. Fundamentally, Taguchi's method and similar treatments of design robustness can be treated as a parametric optimization [11]. Several sources in the literature use robust optimization in order to account for robustness in design [26–29].

The works by Ross, Rhodes, and Hastings have had a noticeable influence on the literature related to robust design quantification [17, 30]. Ross et al. break up robustness into passive robustness and changeability-enabled robustness [17]. Passive robustness is the ability of a design to remain highly valued across various potential future scenarios without requiring a system change [17, 31, 32]. The future scenarios are an ordered set of changing exogenous variables that may affect the design value, e.g., financial situations and operational plans [17]. Many of the quantifications rely on the Fuzzy Pareto Number (FPN) [17]. The FPN is the percent K of the range from the Pareto frontier with respect to cost and utility, i.e., $\text{FPN}(d) = \min\{K | d \subset P_K\}$. The robustness of a design is calculated as the portion of times the design is Pareto efficient across future scenarios. This is referred to as fuzzy Normalized Pareto Trace (fNPT).

Changeability-enabled robustness is the ability of a design to regain its value through a system change after entering a new scenario [17, 32]. The fNPT can be used to measure changeability-enabled robustness by considering the design's Pareto efficiency after the system has undergone change. This is referred to as the Effective NPT (efNPT). Additionally, Fuzzy Pareto Shift (FPS) accounts for the design's utility gained from a system change. FPS is defined as the FPN difference of a design between the unchanged system and the changed system, i.e., $\text{FPS}(d) = \text{FPN}(d) - \text{FPN}(d^*)$, where d^* is the changed system. Finally, the available rank improvement (ARI) measure is used to evaluate the value of a system change mechanism, r. This is defined as the design's maximum possible improvement in utility rank-ordering within the design set using the change mechanism, i.e., ARI $(r, d) = \text{Rank}(d) - \min\{\text{Rank}(d^{r})\}$.

The robustness of a design has been defined as the ability of a design to remain the best among a set of designs as assumptions or noise variables are changed. However, little was found in the literature that gives a direct measure of a system's relative value to other design alternatives. Connelly et al. quantify robustness as the range of a design's rank with respect to the set of designs for a variety of scenarios [33]. For example, a design that ranges from a rank of 1st to 5th across the defined scenarios is deemed more robust than a design that ranges from 1st to 15th. Additionally, for a design that ranges from 5th to 10th, the design would be considered more robust if the baseline as 10th rather than if the baseline was 5th.

The most popular graph theoretic measures are the average path length (APL) and the largest connected component (LCC). Two papers, Sha & Panchal and Kim & Anderson, use both of these measures as an indicator of network robustness [34, 35]. Kim and Anderson use the APL and LCC measures without modification as indicators of robustness. Sha and Panchal compare the robustness of various networks against node failure. As nodes fail, the APL of the network begins to increase until it reaches a maximum after which it rapidly approaches zero. They use the number of failed nodes at which the APL is greatest as the point of comparison of network robustness. Additionally, as nodes fail, the LCC of the network drops. They use the number of failed nodes for which the network's LCC is 2% of its initial value as the point of comparison of network robustness.

Four other quantifications of graph theoretic measures were found in the literature. First, Pape et al. represented a SoS as a network and assessed the robustness of the SoS as the number of SoS interfaces divided by the maximum possible interfaces, where a larger value indicates greater robustness [36]. Second, Adler and Dagli quantify robustness of a network as the average number of node failures due to a cascading failure of nodes [37]. Third, Agarwal et al. use the second smallest eigenvalue of the Laplacian, known as algebraic connectivity. The Laplacian is defined as the difference between the degree matrix and the adjacency matrix. Agarwal et al. apply their quantification to a SoS, where the vertices are the systems and the edges are the interfaces between systems [38].

72.4 Interoperability

72.4.1 Interoperability Definition

Definitions of interoperability are not often stated explicitly within the scope of our literature survey. Instead, a majority of the documentation defines interoperability as the result of following a particular set of steps, standards, models, or frameworks to be followed or implemented by the systems developers or operators. For example, Mordecai and Dori discuss interoperability as the result of system interconnectivity, interfaces, integration, and interaction. These authors attained interoperability through the use of the Systems Modeling Language (SysML) in conjunction with the unified profile for DoDAF and MODAF (UPDM). Similarly, Bowen et al. utilize the levels of conceptual interoperability model (LCIM) developed in 2008 by Tolk et al. The LCIM is an interoperability scale for systems architectures, where each level of interoperability comes with a new set of requirements [39, 40]. There are other standards, frameworks, and active areas of research that explore concepts of interoperability. Regardless of domain and context, all literature within our scope emphasizes that interoperability is achieved through rigorous definition and modeling of systems' behavior and their interactions [41, 42].

A small subset of surveyed documentation does define or discuss interoperability as a technical or sociotechnical concept. Three common themes emerge for definitions of interoperability. First, for software and/or hardware intensive systems, interoperability is defined loosely as the ability to exchange information between entities [39, 43]. Second, for sociotechnical systems, interoperability requires more than just the exchange of information; emphasis is placed on the ability of a system to accomplish one or more emergent behaviors during operations [44–46]. Third, several authors explicitly note the complicated nature of interoperability and claim that no unitary definition will ever exist [47–49]. For example, Agarwal et al. investigate interoperability with respect to net-centricity and note that interoperability may have multiple complimentary dimensions; these include sharing an interface, sharing semantically compatible data or resources, and guaranteeing operational compatibility [50].

72.4.2 Interoperability Quantification

Many of the definitions discussed in the above section of this paper may be thought of as a specialized type of quantification, where a system or set of systems may have a Boolean value for interoperability if a certain process is followed. In-depth discussion on this class of quantification is excluded from this paper for brevity. Some interoperability models and frameworks do provide quantitative measurement for interoperability, implementing a scale by which engineers may better assess their systems at different phases in the lifecycle. For more information, the reader is referred to Wyatt et al., who conducted a comprehensive assessment on six prominent interoperability models that were created between 1980 and 2012 [41]. Nonetheless, several explicit quantifications were found for interoperability in the literature review. These quantifications draw from research in complexity theory, reliability theory, and net-centric systems.

Haghnevis and Askin describe an integrated framework to study emergent behavior, the consequence of evolution, and adaptation in complex adaptive systems (CAS) [51]. A portion of the framework is dedicated to interoperability, through which system emergence is assessed and analyzed. Consider a collection of components within a CAS; they may cooperate/compete against one another according to one or more patterns of behavior. For example, consumers of electricity (components) using the national power grid (CAS) may consume electricity according to different patterns of behavior. Consumption may depend on factors such as location, price of service, and time of year. The patterns of electricity consumption, as well as their ability to evolve over time, can be considered an integral part of the understanding CAS itself. The interoperability of a CAS may then be defined and quantified through its constituent patterns, ($i = 1 \ldots n$), that a population of components, X_i, will adhere to. Equation 72.1 measures the interoperability, I, between patterns i and j. $P_{m_i m_j}$ represents the joint probability to simultaneously find pattern i and j in the same states of behavior m_i and m_j, while M represents the total of states. Similarly, P_{m_i} is the marginal property to find pattern i in state m_i. The value of interoperability may then be precisely defined as the amount of information that may be exchanged between patterns of behavior i and j, which may range from 0 (if no states m are shared) to 1.

$$I^P(i;j) = \sum_{m_i=1}^{M_i} \sum_{m_j=1}^{M_j} P_{m_i m_j} \log_2 \frac{P_{m_{ii} m_j}}{P_{m_{ii}} P_{m_j}} \tag{72.1}$$

A third pattern of behavior, k, is then considered in conjunction with the pair discussed above, i.e., $I(i;j;k) = I(i;j) - I(i;j|k)$ [51]. Patterns such as k are defined as positive or negative catalysts of interoperability with respect to pattern pair i and j, depending on their impact to the value in Eq. 72.1. The catalyst-associate interoperability (CAI) measure represents the total impact k has on all pattern pairs in the system, μ, i.e., $\mathrm{CAI} = I(\mu \mid k) - I(\mu)$ [51]. The authors propose that higher values for interoperability between patterns indicate less autonomy and system emergence, while lower values of interoperability describe highly autonomous components/CAS [51].

Drawing from reliability theory, Wyatt et al. develop a metric for interoperability that is utilized during the conceptual design phase. The directional value of interoperability for a resource exchange from system i to system j is given as the product of the transaction's reliability of transmission, Θ_m, and the reliability of translation, Θ_l, i.e., $\Theta_{ij} = \Theta_l \Theta_m$. Reliability of translation decomposes into P_l, the probability that the resource will be translated, and τ_q, the quality of translation. The quality of translation will be between 0 and 1, and can be thought of as a probability of correct translation. The equation $\Theta_l = P_l(\tau_q) + (1 - P_l)$ combines these concepts; a given interaction's reliability of translation can have a maximum value of 1 (if $P_l =$ 0) and a minimum value of τ_q (if $P_l = 1$) [52].

For each resource exchange, Wyatt's metric will be used to build into an n x n resource transfer interoperability matrix, where n is the number of system types in the architecture. If a pair of systems conducts more than one resource exchange, the element-wise multiplication of all above-mentioned matrices will result in the systems of systems interoperability matrix. This matrix may then be utilized by other external analysis models such as ARCNET, which assesses the benefits between increased resource exchange and mission effectiveness [47].

Agarwal et al. investigate several analysis techniques and their applicability to net-centric SoS architecture analysis. The authors posit that a critical attribute of a given architecture is its net-centricity and its ability to share information across constituent systems [48]. Net-centricity is decomposed into two concepts: interoperability and communication. Interoperability is defined as the ability to share an interface with another constituent system, while communication determines whether or not a set of participating systems are coordinating their operations through common communication channels [48]. The measure of interoperability is given as a summation of interface connections for a network, i.e., $\sum_{i=1}^{N}\sum_{j=1}^{N} A_{ii}A_{jj}A_{ij}$. The variable $\boldsymbol{A}$ represents a binary adjacency matrix, with values of 1 indicating the existence of an interface between systems i and j. N represents the total number of participating systems in any given architecture configuration. A separate summation is used for the measure of communication, i.e., $\sum_{k=1}^{M} \phi A_{ki}A_{ik}$. The variable M is defined in this publication as the total number of communication channels k, where a nonzero value in A represents the participation of both i and j over channel k. The value ϕ represents scaling factor for each potential communication channel, a positive value may represent an increased incentive for use, while a negative value may indicate degradation or insecurity. The value of net-centricity for a given SoS architecture is then the summation of both equations, and it is used as an attribute in a genetic algorithm to better assess alternative architectures earlier in conceptual design.

72.5 Agility

72.5.1 Agility Definition

The literature review uncovered several common themes among the definitions of agility in a systems-engineering context. Despite the general agreement within the literature, a single commonly cited definition has not emerged. These findings match that of Ryan et al., who performed a literature review of “flexibility-related terminology” in 2013 [9]. This section discusses the common themes that arise when defining agility, as well as specific definitions relevant to the quantification of agility.

Before reviewing the definitions of system agility, it is important to note the distinction between an agile system and an agile systems-engineering method. An agile system is a system that exhibits agility, whereas agile systems-engineering methods employ agile principles during the systems-engineering process. The former is the subject of greater interest to this paper, because it is the quantification of system agility that is under examination. Dove & LaBarge provide a thorough review of the difference between the two topics; Asan, Carlson & Turner, Necaille, and Pamujula et al. provide discussion on the use of agile systems-engineering methods [53–58].

The fundamental principle found in all definitions of system agility is the ability of the system to change. Commonly added to that is the notion that the system can undergo change “rapidly” [59–62]. Additionally, some authors specifically state that the system is in an “uncertain” or “unpredictable” environment [53, 63, 64]. Others, such as Bauer et al. and Silva et al., leave out any mention of the environment [60, 61]. Ryan et al. propose a succinct definition based on their literature review, that is, in general agreement with the themes discussed above. They state that “agility is the measure of how quickly a system’s capabilities can be modified in response to external change” [9]. However, the definition is prefaced by two important and still relevant points: there remains insufficient academic material to reach a consensus definition of agility, and the definitions that do exist are remarkably similar to definitions provided for flexibility [9]. The second point matches the findings of our previous work, which was unable to find a consensus definition of flexibility. Instead, the definition of flexibility also revolved around a system’s ability to change, with additional criteria discussing the varying circumstances leading to change [10].

Two definitions specifically referenced enterprise agility and manufacturing agility, with corresponding quantifications discussed in the following section. Brown states that enterprise agility is “the ability of the Enterprise to respond to the rate of change of its Environment and adapt to the new Environment accordingly” [65]. Jung et al. propose that “agility enables the manufacturing system to shorten the time to recovery while also maintaining a high level of residual performance during the disturbance” [66]. Although the latter is not a traditional definition, both it and Brown’s definition include the core principle which is the ability to change. Likewise, Jung et al. explicitly mention the dependence on time, whereas Brown’s definition implicitly includes time by describing the rate at which the enterprise can change.

72.5.2 Agility Quantification

Only two papers were found that discuss metrics to quantify system agility; two additional sources were found that propose general metrics relating to system agility that could be quantifiable. The common theme between all four papers is the dependence of agility on time.

Papers by Brown and Harris (both of The SI Organization, Inc.) reference the “Enterprise Agility Equation,” given by Eq. 72.2, which states that the ability of the enterprise to change its capabilities must be greater than the environment’s ability to change, if it is going to survive [65, 67].

$$\frac{\mathrm{d}}{\mathrm{d}t}\text{Enterprise Capability} > \frac{\mathrm{d}}{\mathrm{d}t}\text{Environment} \tag{72.2}$$

Harris suggests that the performance of an enterprise is the best way to capture its response to a changing environment [67]. Furthermore, if the performance of an enterprise is measured with respect to time, agility can be calculated as the area under the curve after the enterprise has undergone a disturbance resulting in a performance degradation. That area must occur within a time limit known as the duration of need. The duration of need is constrained by how long the performance is required by the enterprise. Recovery outside of the duration of need is not necessarily as useful to the enterprise, because the demand for performance has passed [67].

Although Harris et al. do not give explicit examples of enterprise performance metrics, they note that the metrics should be chosen carefully such that they can realistically be measured over time, and the values of the metrics can be improved through enhancements to the enterprise. A method is also described that implements enterprise and environmental simulations in a gaming framework, which allows one to model the interaction between the two entities. The virtual modeling environment is used to measure the value the enterprise provides in response to various inputs. This corresponds to the necessity of the enterprise to change in response to a changing environment, which is the concept behind the enterprise agility equation, given by Eq. 72.2. Ideally, this framework can then be used to shape the development of, and future enhancements to, the enterprise [67].

Jung et al. present the agility of a manufacturing process as a function of both recovery time and residual performance. Recovery time is defined as the time between the initial decrease in performance and the time at which the initial level of performance is regained. Residual performance is defined as the percent of perfect orders successfully filled during the disturbance. The agility of the system can be maximized by minimizing the time to recovery and maximizing the residual performance.

Finally, two papers propose general metrics that can be used to measure agility. Necaille quotes the definition of agility relating to network enabled capabilities as "the ability to respond to changing circumstances," where it can be measured through speed of action, cost in resources, and impact on effectiveness [58].

Dove & LaBarge assert that "agility does not have a practical absolute measure," and instead propose that response proficiency can be used to characterize a system's agility [53]. Response proficiency, corresponding to both proactive and reactive changes in response to a system's changing environment, is then broken down into four metrics: time, cost, predictability, and scope. Time to respond includes any time associated with a decision to respond and the time to actually accomplish that response. Cost of response includes any costs associated with either implementing the response or resulting from the response. Predictability of response corresponds to the ability of the system to repeatedly deliver an effective response. Scope of response corresponds to the system's ability to respond within a certain mission. It is confirmed through the ability to repeatedly accommodate a breadth of response scenarios.

72.6 Conclusions

This paper presented a systematic literature review on the definitions and quantifications of three ilities: robustness, interoperability, and agility. The literature review covered eight conferences and journals between 2010 and 2015. The definitions were found to demonstrate relative agreement for each of the ilities; however, many definitions differed in the details. The prevalence of quantifications varied for each ility. Robustness was found to have large number and variety of quantifications, covering both system robustness and design robustness. A moderate number of quantifications were found for interoperability; however, many were specific to certain applications. Additionally, very few quantifications were found for agility. The authors recommend that further attention from the system-engineering community is needed to develop quantified measures for interoperability and agility as well as demonstrate their application.

Disclaimer The authors' affiliation with The MITRE Corporation is provided for identification purposes only, and is not intended to convey or imply MITRE's concurrence with, or support for, the positions, opinions or viewpoints expressed by the authors. 16-3392

References

1. de Weck OL, Ross AM, Rhodes DH (2012) Investigating relationships and semantic sets amongst system lifecycle properties (Ilities). In: International engineering systems symposium
2. Holland J (2014) Engineered resilient systems. In: NDIA systems engineering conference, D.C
3. Holland J (2015) Engineered resilient systems. In: NDIA 16th science and engineering technology conference, D.C
4. Ross AM, Rhodes DH (2015) Towards a precriptive semantic basis for change-type ilities. Procedia Comput Sci 44:443–453
5. de Bruijn H, de Bruijne M, ten Heuvelhof E (2015) The politics of resilience in the Dutch 'room for the river' -project. Procedia Comput Sci 44:659–668
6. Goerger SR, Madni AM, Eslinger OJ (2014) Engineered resilient systems: a DoD perspective. Procedia Comput Sci 28:865–872
7. Ricci N, Fitzegerald ME, Ross AM, Rhodes DH (2014) Architecting systems of systems with ilities: an overview of the SAI method. Procedia Comput Sci 28:322–331
8. Beesemyer JC, Ross AM, Rhodes DH (2012) An empirical investigation of system changes to frame links between design decisions and ilities. Procedia Comput Sci 8:31–38
9. Ryan ET, Jacques DR, Colombi JM (2013) An ontological framework for clarifying flexibility-related terminology via literature survey. Syst Eng 16:99–110
10. Turner AJ, Cotter M, Monahan W (2016) Quantifying the ilities: a literature review of resiliency, flexibility, and adaptability. In: Conference on systems engineering research, Huntsville
11. Malak R, Baxter B, Hsiao C (2015) A decision-based perspective on assessing system robustness. In: 2015 conference on systems engineering research, Hoboken
12. Sitterle V, Freeman D, Goerger S, Ender T (2015) Systems engineering resiliency: guiding tradespace exploration within an engineered resilient systems context. Procedia Comput Sci 44:649–658

13. Pape L, Dagli C (2013) Assessing robustness in systems of systems meta-architectures. In: Complex adaptive systems
14. Sols A (2015) Validation of the robustness of the selection of the preferred design concept through a comprehensive sensitivity analysis. In: 9th annual IEEE international systems conference (SysCon), 2015
15. Squires A, Cloutier R (2011) Applying a robust design approach to improve online systems engineering education. In: IEEE international systems conference (SysCon)
16. Georgiadis DR, Mazzuchi TA, Sarkani S (2013) Using multi criteria decision making in analysis of alternatives for selection of enabling technology. Syst Eng 16(3):287–303
17. Fitzgerald ME, Ross AM (2012) Mitigating contextual uncertainties with valuable. In: IEEE international systems conference, Vancouver
18. Pape L, Agarwal S, Giammarco K, Dagli C (2014) Fuzzy optimization of acknowledged system of systems meta-architectures for agent based modeling of development. In: 2014 conference on systems engineering research
19. Acheson P, Pape L, Dagli C, Kilicay-Ergin N, Columbi J, Haris K (2012) Understanding system of systems development using an agent- based wave model. Procedia Comput Sci 12:21–30
20. Guariniello C, DeLaurentis D (2014) Communications, information, and cyber security in systems-of-systems: assessing the impact of attacks through interdependency analysis. In: 2014 conference on systems engineering research, Redondo Beach
21. Guariniello C, DeLaurentis D (2014) Integrated analysis of functional and developmental interdependencies to quantify and trade-off ilities for system-of-systems design, architecture, and evolution. In: 2014 conference on systems engineering research, Redondo Beach
22. Gomez M, Kim Y, Matson E, Tolstykh M, Munizzi M (2015) Multi-agent system of systems to monitor wildfires. In: 10th system of systems engineering conference (SoSE)
23. Muller G (2012) Fuzzy architecture assessment for critical infrastructure resilience. Procedia Comput Sci 12:367–372
24. Dickerson C, Mavris D (2013) A brief history of models and model based systems engineering and the case for relational orientation. Syst J 7(4):581–592
25. Gomez M, Kim Y, Matson E, Tolstykh M, Munizzi M (2015) Multi-agent system of systems to monitor wildfires. In: System of systems engineering conference (SoSE)
26. Mour A, DeLaurentis D (2014) Bandwidth allocation in tactical data links via mechanism design. In: Conference on systems engineering research
27. Shindin E, Boni O, Masin M (2014) Robust optimization of system design. In: Conference on systems engineering research
28. Davendralingam N, DeLaurentis D (2015) A robust portfolio optimization approach to system of system architectures. Syst Eng 18(3):269–283
29. Davendralingam N, Kenley R (2013) A mechanism design framework for the acquisition of independently managed systems of systems. In: International conference on system of systems engineering, Maui
30. Keane A, Gaspar H, Brett PO (2015) Epoch era analysis in the design of the next generation offshore subsea construction vessels. In: 10th system of systems engineering conference (SoSE)
31. Ross A, Rhodes D, Hastings D (2008) Defining changeability: reconciling flexibility, adaptability, scalability, modifiability, and robustness for maintaining system lifecycle value. Syst Eng 11(3):246–262
32. Ross A, Rhodes D, Hastings D (2009) Using pareto trace to determine system passive value robustness. In: IEEE international systems conference. Vancouver
33. Connelly E, Colosi L, Clarens A, Lambert J (2015) Analysis of biofuels industry for aviation with scenario-based expert elicitation. Syst Eng 18(2):178–191
34. Sha Z, Jitesh PH (2013) Towards the design of complex evolving networks with high robustness and resilience. In: Conference on systems engineering research

35. Kim H, Anderson R (2013) An experimental evaluation of robustness of networks. Syst J 7 (2):179–188
36. Pape L, Giammarco K, Colombi J, Dagli C, Kilicay-Ergin N (2013) A fuzzy evaluation method for system of systems meta-architectures. In: 2013 conference on systems engineering research
37. Adler CO, Dagli CH (2014) Study of the use of a genetic algorithm to improve networked system-of-systems resilience. Procedia Comput Sci 36:49–56
38. Agarwal S, Pape LE, Dagli CH (2014) A hybrid genetic algorithm and particle swarm optimization with type-2 fuzzy sets for generating systems of systems architectures. In: Complex adaptive systems, Philadelphia
39. Bowen R, Sahin F (2010) A net-centric XML based system of systems architecture for human tracking. In: Systems of systems engieering conference (SoSE)
40. Bowen R, Sahin F (2013) Net-centric system of systems framework for human detection. In: Systems of systems engineering conference (SoSE)
41. Wyatt E, Griendling K, Mavris E (2012) Addressing interoperability in military systems-of-systems architectures. In: Systems conference (SysCon), 2012 I.E. international
42. Mordecai Y, Dori D (2013) A model-based framework for architecting system-of-systems interoperability, interconnectivity, interfacing, integration, and interaction. In: INCOSE international symposium
43. Wacholder D, Stary C (2015) Enabling emergent behaviour in systems-of-sytems through bigraph-based modeling. In: System of systems engineering conference (SoSE)
44. Fry D, DeLaurentis D (2011) Measuring net-centricity. In: 2011 6th international conference on system of systems engineering (SoSE), pp 264–269
45. Madni A, Sievers M (2014) Systems integration: key perspectives, experiences, and challenges. Syst Eng 17
46. Vaneamn W, Jaskot R (2013) A criteria-based framework for establishing system of systems governance. In: Systems conference (SysCon), 2013 I.E. international
47. Domercant J, Mavris D (2012) ARCNET: a system-of-systems architecture resource-based collaborative network evaluation tool. In: Systems of systems engineering conference (SoSE)
48. Agarwal S, Pape L, Dagli C (2014) A hybrid genetic algorithm and particle swarm optimization with type-2 fuzzy sets for generating systems of systems architectures. Procedia Comput Sci 36:57
49. Moschoglou G, EveLeigh T, Holzer T, Sarkani S (2012) A semantic mediation framework for architecting federated ubiquitous systems
50. Agarwal S, Pape LE, Dagli CH, Ergin NK, Enke D, Gosavi A, Qin R, Konur D, Wang R, Gottapu RD (2015) Flexible and intelligent learning architectures for SoS (FILA-SoS): architectural evolution in systems-of-systems. Procedia Comput Sci 44:76–85
51. Haghnevis M, Askin R (2012) A modeling framework for engineered complex adaptive systems. Syst J 6:520
52. Wyatt E, Domercant JMD (2013) A reliability-based measurement of interoperability for systems of systems. In: Systems conference (SysCon)
53. Dove R, LaBarge R (2014) 8.4.1 fundamentals of agile systems engineering - part 1. In: INCOSE international symposium
54. Dove R, LaBarge R (2014) 8.4.2 fundamentals of agile systems engineering - part 2. In: INCOSE international symposium
55. Asan E (2015) Improving agility by knowledge driven and collaborative systems of systems engineering. Int J Syst Syst Eng 6(3):186–220
56. Carlson R, Turner R (2013) Review of agile case studies for applicability to aircraft systems integration. In: 2013 conference on systems engineering research
57. Pamujula S, Thoppay R, Kelly K, Fish R (2012) 11.4.3 impact of agile process in systems engineering. In: INCOSE international symposium
58. Necaille C (2011) 6.3.1 process patterns for agile capability engineering methodology: the PACEM project. In: INCOSE international symposium

59. Asan E, Bilgen S (2012) Agile collaborative systems engineering -motivation for a novel approach. In: INCOSE international symposium
60. Bauer W, Elezi F, Roth M, Maurer M (2015) Determination of the required product platform flexibility from a change perspective. In: Systems conference (SysCon), 2015 9th annual IEEE international
61. Silva E, Sa R, Ribeiro J, Sabaleuski A, Nascimento F, Faro T, Simoes R (2012) A practical framework for the generation and selection of strategic plans of action: the relevance of timeframe considerations and a systems thinking approach. In: Systems conference (SysCon), 2012 I.E. international
62. Stelzmann E (2011) Contextualizing agile systems engineering. In: Systems conference (SysCon), 2011 I.E. international
63. Schapiro S, Henry M (2012) Engineering agile systems through architectural modularity. In: Systems conference (SysCon), 2012 I.E. international
64. Zonnenshain A (2013) Agile and lean SE: are these two approaches complementary or they are different and should applied in specific environment? In: INCOSE international symposium
65. Brown DE (2013) The agile enterprise: systems engineering agility at the enterprise level. In: INCOSE international symposium
66. Jung K, Morris K, Lyons KW, Leong S, Cho H (2015) Mapping strategic goals and operational performance metrics for smart manufacturing systems. In: Conference on systems engineering research
67. Harris P (2013) 4.2.2 a framework and metrics for addressing an agile enterprise. In: INCOSE international symposium

Chapter 73
A Systems Integration Framework for Interdisciplinary Black Sky Operations

Ellie Graeden and Joel Thomas

Abstract During a large-scale "Black Sky" power outage in the continental United States, the private sector as well as federal, state, local, tribal, and territorial emergency management communities will be tasked with providing support for an extended period of time to a very large population over a large geographic region. Such an event would cause unprecedented essential service disruptions not only within the power sector, but also for transportation, water, and communications sector infrastructure, as well. Most critically, any communications systems that can operate during such an emergency will be limited in bandwidth (compared with the communications systems that we all use every day), potentially causing an adverse effect on cross-sector coordination efforts. As a result, there is a need to preplan and develop, in advance of such a "Black Sky" power outage, an appropriate human and technical interoperability framework that will support the eventuality of a Black Sky event, and thereby pick out the most-critical data to be generated and shared, so as to be able to conduct restoration and emergency-management activities within the available bandwidth.

To help address the need to develop a communications system that will function during Black Sky events, we have used a systems analysis approach to research and identify usage requirements for the purposes of command, control, coordination, and communications. Black Sky operational mission requirements were identified across multiple disciplines for the steady-state, response, and recovery phases of emergency management. Each operational requirement was cross-walked to the corresponding information requirements needed to support the activity. Black Sky operational mission requirements were categorized as fulfilling one of three mission-critical functions: execution of strategic mission priorities; cross-sector planning, and coordination; or resource request and acquisition. The corresponding information requirements were

Electronic supplementary material: The online version of this chapter (doi:10.1007/978-3-319-62217-0_73) contains supplementary material, which is available to authorized users.

E. Graeden (✉)
Talus Analytics, Lyons, CO, USA
e-mail: egraeden@talusanalytics.com

J. Thomas
SPIN Global, Washington, DC, USA
e-mail: jthomas@spinglobal.co

organized by utility of the information type: event characterization; consequence analysis; and/or decision support. Data elements were then aligned with each information requirement to define the most appropriate and comprehensive source of that information. Each data element was described by the relative load requirements for information transfer within a Black Sky-functional communications system.

Based on the results of this analysis, we defined the relative total load requirements for a communications system designed to fulfill the information needs of the user community during a Black Sky event. The results have been aligned to the systems analysis performed; this structure will help ensure that responders have the right information, at the right time, to effectively perform their missions and guide alignment of the information needs to the Black Sky communications system infrastructure. This analysis is the first to connect and prioritize the operational activities of the emergency management community to the technical information, data, and communications systems load for large-scale power outages.

Keywords System architecture • Black Sky • Information sharing • Emergency communications

73.1 Introduction

Research and real-world events have shown that the existing communications system in the United States cannot be relied upon in the aftermath of a large-scale emergency, as it would likely be unavailable for use. This loss poses a tremendous problem, as the Electric Infrastructure Security (EIS) Council has demonstrated through its Black Sky exercises: the lack of effective and reliable electronic communications is one of the critical factors in the near-catastrophic severity of such events. In response to these previous findings, EIS is working to develop an emergency communications system (ECOM) to fill this gap for a large-scale Black Sky event triggered by natural or man-made causes.

The requirements for such an ECOM have previously been based on expert analysis drawn largely from a command and control system, as used by the U.S. military. In command and control environments, the information requirements include the situational awareness data required to understand the event as it evolves, as well as the information required to manage and perform restoration. By contrast, within the United States, the emergency management community is tasked with coordination and communications functions, while restoration activities are performed by private and public utilities. Therefore, the technical demands of an ECOM, as envisioned by EIS, requires an understanding of the information requirements to support coordination and communications between players, but it is assumed to be limited in the requirements focused on restoration activities.

Here, we present the results of an analysis of the information requirements of an ECOM to support emergency response and restoration activities following a Black Sky event, as described previously by EIS. The results are organized by operational, information, and data requirements for multisector coordination and communications. Associated metadata were defined to support the technical specifications for an ECOM, as described in a previous report [1], and to inform development of a companion software-based national recovery coordination system, described in the remainder of this report as BSX, that will facilitate command, control, communications, coordination, and counterintelligence [2].

73.2 Methods

The following details the systems analysis methods, coordination emphasis, operational phases, and information management methods. The overall goal of the research and analysis was to better understand usage requirements of the BSX system for the purposes of command, control, coordination, and communications and to design an implementable framework that would inform workflow and load requirements.

73.2.1 Systems Analysis Approach

To identify and categorize the information requirements for an ECOMs for Black Sky events, we used a systems analysis approach incorporating analysis from emergency management at the federal, state, and local levels; previous analysis of the information available to these types of decision makers; and information available from and within the energy sector for the steady state, response, and recovery phases of emergency management. Results from previous research and experience working in emergency management suggest that information requirements are directly tied to mission-specific operational tasks, each supported by specific data [4, 11, 14–16]. A systems-level framework was developed to define and link these operational mission requirements (tasks) to the information requirements needed to inform the operational mission, and, in turn, to the data needed to meet the information requirement.

Operational mission requirements were defined through a series of 12 regional information-sharing projects sponsored by a U.S. Department of Homeland Security research initiative known as "Virtual USA" [5, 6]. During program implementation, ~220 total mission requirements were collated from ~4500 individuals and ~50 working groups representing over 500 state and local agencies throughout 40 states, 7 Canadian provinces, including 25 state's National Guard, and more than 150 private sector companies and nongovernmental organizations. This effort included the analysis and prioritization of strategic mission priorities by State Emergency Management Directors, Adjutants General, and other senior government officials, often resulting in a much smaller sample (i.e. less than two dozen) of operational mission requirements to support the functions of command, control, coordination, and communications across multiple jurisdictions. The researchers' understanding of operational mission requirements was refined through prior energy sector emergency management research [4], including conversations with emergency management personnel from relevant federal agencies, and on the priorities outlined in the National Response Framework and its Emergency Support Function annexes [7]. The operational mission requirements include those specific to each phase described in the National Response Framework FIOP and specific to Black Sky events, even when not previously prioritized.

Information requirements were defined based on use case analysis of the specific user groups previously referenced, defined as “essential elements of information” (EEIs) required to support a specific operational mission requirement(s). The information requirements were aligned with the National Strategy for Information Sharing & Safeguarding [8] and the Information Interoperability Framework [9]. In developing the BSX information requirements, 350+ EEIs were consulted from research of the Virtual USA program that have been refined by the Incident Management Information Sharing Sub-Committee (IMIS-SC) [10] of the White House National Security Council. Our final analysis focused on those information requirements that directly support the operational mission requirements of the target user communities in a Black Sky event and defined based on previous analyses [4, 11, 14–16]. The results include information identified as needed to support operational mission requirements during events by experts in emergency management response within the energy sector.

Data requirements were defined based on a series of 350 interviews with 458 individuals across the federal interagency emergency management community and linked to corresponding information requirements, as defined above, and through the work of the IMIS-SC [10, 11, 14–16]. Data requirements are defined by the datasets needed to meet each information requirement (e.g., infrastructure of concern in the impacted area, population with durable medical equipment). Each data requirement is characterized by the relative information transfer load within a Black Sky-functional communications system, as defined by the number and type of fields required for tabular data and text and the number and type of geospatial elements and metadata fields required for maps. These results define the technical specifications required to meet the information needs of the emergency management community during a Black Sky event, ensuring that responders have the right information, at the right time, to effectively perform their missions.

73.2.2 Emphasis on Coordination

Our research and analysis builds upon the understanding that response and recovery for a Black Sky emergency, supporting the “whole of community,” requires an information management [9] and communications interoperability [12] framework to serve the operational coordination and management functions of the emergency management community at federal, state, local, tribal, and territorial scales. The primary management requirement during a Black Sky event, or any other emergency, is to understand the situation, what tasks are required to mitigate losses, and the process by which the response and recovery efforts can most effectively and efficiently mitigate those losses. The primary coordination task is to determine how organizations can best work together to coordinate prioritization and delivery of services to support the needs of the “on-the-ground” operational response and recovery apparatus of private and public organizations, given the dependencies between the sectors and between each task of response and recovery.

This approach explicitly recognizes the roles and responsibilities of each organization and asset owner to execute their own Black Sky emergency plans within their respective chains of command, while addressing the need for national-level coordination of activities. This coordination across the emergency management community involves multiple sectors and requires macro-level coordination throughout the community within the context of command, control, coordination, and communications (C4). Notably, this approach posits that macro-level coordination may coexist with micro-level coordination, defined as that which is managed from within an organization that may have greater tolerance for traditional C2 approaches, as typically employed by the military in theater. Therefore, the approach proposed here to effectively manage and coordinate Black Sky emergency response and recovery includes defining clear operational mission requirements that support the functions of management and coordination, and supporting information and data requirements that serve the mission of national, regional, and local efforts. This approach is designed to support a nimble communications architecture that will provide the information needed for effective management and coordination within a “Whole of Community” effort, while relying heavily upon distributed responsibility for response and recovery activities by asset owners and locally managed organizations in widely disparate geographic regions throughout the United States.

73.2.3 Operational Phases

The Black Sky Operational Phases (Fig. 73.1) were drawn directly from the Federal Emergency Management Agency (FEMA) Response Federal Interagency Operational Plan (FIOP). The FIOPs are built upon the concepts outlined in the National Response Framework and serve as operational documents specifying how various federal agencies work and interact to support national preparedness. There are five FIOPs – each describing one of the preparedness mission areas – prevention, protection, mitigation, response, and recovery [3].

The National Response Framework and the five FIOPs articulate a comprehensive vision of how the numerous agencies comprising the preparedness community can work together using common language and operational procedures, thereby aligning their mission-specific practices with those of the overall community. In support of this integration strategy, three emergency management phases and eight subphases used in this document are drawn directly from the FEMA Response FIOP.

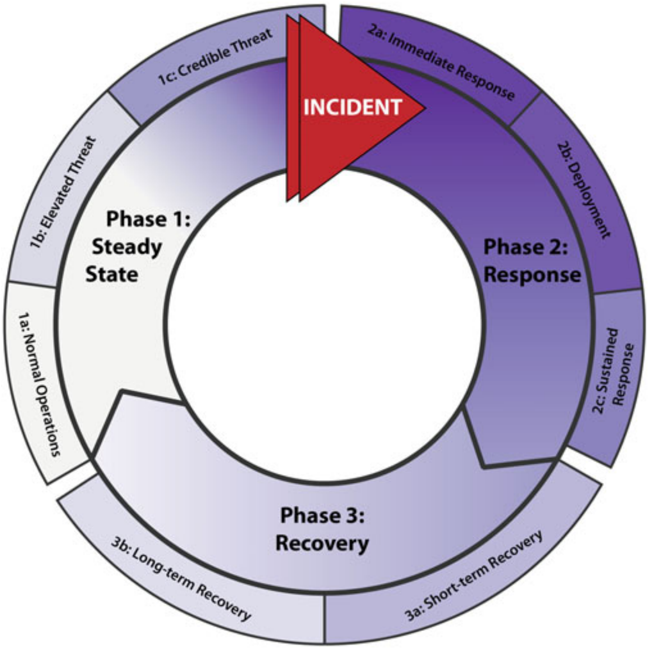


Fig. 73.1 Response FIOP phases of emergency management

73.2.4 *Information Management*

For each Black Sky Operational Subphase, a series of three information categories was provided to broadly scope and define the related operational information and data requirements. These three information categories are event characterization, consequence analysis, and decision support [11].

Event characterization models and analyses convert raw observational data into situational awareness information describing the location, timing, and/or severity of an event. Event characterization performed during a hazardous event may predict, for example, the cities or regions likely to be affected and to what degree. Event characterization may occur before, during, or following an event to support long-term planning of a hypothetical event, rapid assessment of an ongoing event, or extent validation of an event that has already occurred, respectively.

Consequence analysis incorporates infrastructure, economic, and/or population data into the extent, timing, and severity results produced via event characterization. Consequence analysis produces impact estimates for the affected areas, including, for example, economic loss, infrastructure damage, and disruptions to supply chains. During a hazardous event, consequence analysis may be performed following event characterization to estimate, for instance, the potential number of affected or displaced individuals, proportion of buildings damaged, human health effects, and/or economic consequences in the identified high-risk regions.

Decision support analysis translates the impact estimates produced by consequence analysis into actionable, mission-specific data. Decision support analysis produces estimates such as the amount of materials or number of personnel needed to support each activity. During an event, consequence analysis may predict the number of individuals likely to be affected by a hazard, and then decision support analysis may be performed to predict, for example, the number of medical professionals and supplies needed to treat those individuals. Consequence and decision support analysis have significant utility before and during an event, as they allow emergency personnel to predict and mobilize the appropriate resources before a hazardous event has reached a catastrophic level.

Furthermore, the information requirements presented detailed at a minimum the following:

- Who will use the information;
- Name of the information item;
- Description of the information item;
- Is this information item likely to be *required*, or only *desired;*
- Form the information item will take;
- Frequency of measurement or acquisition required to support the mission;
- Accuracy/quality requirements;
- Source(s) for this information item; and
- Rationale for the need/desire for this information item.

73.3 Results

A systems analysis approach was used to define and structure the results of our research and analysis into collections of operational, information, and data requirements for the BSX.

73.3.1 Systems-Level Overview

A relational database was constructed based on the results of the systems-level analysis of Black Sky requirements described previously (Fig. 73.2). The database includes three requirements tables focused on operational missions, information requirements, and the corresponding data requirements for each. Metadata attributes (bulleted lists in Fig. 73.2) describe each requirement's role and technical characteristics in the context of Black Sky operations, as relevant to the BSX.

Like the systems framework, the relational database is hierarchical, with operational mission requirements fed by one or more information requirements and information requirements fed by one or more data requirements. These interconnections are specified using an associative table of keys that defined which

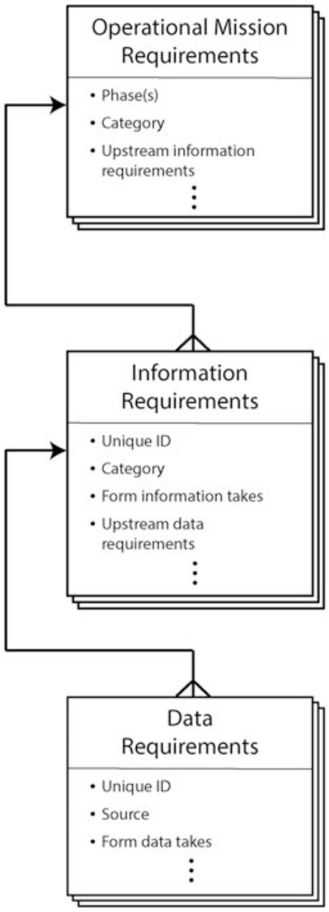


Fig. 73.2 Systems-level framework for Black Sky requirements

information requirements fed each operational mission requirement and which data requirements feed each information requirement.

A set of 34 operational mission requirements, 16 information requirements, and 26 data requirements are described (see Fig. 73.2 and Appendices A, B, and C). The requirements were developed based on the research team's analysis and experience in supporting the function of emergency management and from Virtual USA projects, conversations, and interviews with professionals in related communities of practice.

73.3.2 Requirements Databases

Three distinct and interconnected requirements databases were developed based on the systems-level framework: operational mission requirements; information requirements; and data requirements.

Table 73.1 Example operational mission requirement and metadata attributes

Phase(s)	Operational mission requirements	Category	Description	Upstream information requirements
Deployment	Transport resources into impacted area	Resource request and acquisition	Transport the personnel, meals, water, utility repair equipment, and other resources that are needed for the response into the impacted areas, using the most direct functional ingress routes from the resource sources.	I9, I10, I11, I15

73.3.2.1 Operational Mission Requirements

Operational mission requirements define phase-specific actions taken by the emergency management coordinating entity. The operational mission requirements reflect the core response and recovery activities coordinated by emergency management agencies relevant during a Black Sky event. They are focused on actions required to facilitate the overall progress of the steady state, response, and recovery phases by ensuring effective and timely coordination of information and cross-sector activities. Operational mission requirements are focused on emergency operations and goals overall without specifying individual, organization-level tasks. For example, the operational mission requirement “Transport resources into impacted area” defines the requirement to deliver food, water, and other mass care resources to impacted areas, without stating which agencies or groups are responsible for the task.

An example operational mission requirement is provided in Table 73.1. The complete operational mission requirements table is provided in Appendix A.

73.3.2.2 Information Requirements

Each operational mission requirement is linked to one or more information requirements that defines the information needed by the coordinating entity to effectively carry out operational mission requirements. Information requirements are based on the information that is both available and necessary to guide each operational mission requirement. Most operational mission requirements are supported by multiple information requirements. Information requirements represent the integration of data sources owned by a range of agencies and organizations to provide essential context for response and recovery operations beyond those provided by the individual data elements. For example, the information requirement “Transportation ingress/egress status” synthesized transportation infrastructure status, power outage estimates, fuel availability and supply requirements, and the active hazard event data to provide a map of current transportation status relative to the active hazard footprint. This collated information product can be used to directly support

Table 73.2 Example information requirement and metadata attributes

Unique ID	Information requirement	Category	Information format	Upstream data requirements
I9	Transportation ingress/egress status	Consequence analysis	Map (first layer):for each state, the location of each transportation node/road is mapped, with icon or line color indicating status (open, restricted, closed), and time stamp Map (second layer):contours defining the geographic extent of each severity level of the hazard, using the appropriate hazard-relevant scale (hurricane wind categories, earthquake shaking intensities, etc.), and timestamp Table:transportation node name, type (e.g., bridge, road, port), name, service area (state, county), address, lat/long, status (open, restricted, closed), estimated date and time of reopening, and timestamp	D0, D1, D2, D6, D11, D16

operational mission requirements that concerned moving in and out of the impacted area, such as "Transport resources into impacted area."

An example information requirement is provided in Table 73.2. The complete information requirements table is provided in Appendix B.

73.3.2.3 Data Requirements

Each information requirement is linked to one or more data requirements that defined the specific datasets that needed to be collected and available during Black Sky operations to support response and recovery decision-making. The data requirements are focused on individual datasets that need to be stored and made available on an ECOM and BSX platform to support operational mission requirements. Data requirements are based on the specific, granular data elements needed to create the product described by each information requirement. The relative load requirements for the ECOM and BSX platform are defined by the data fields needed for each data requirement. Data requirements are described as tables, maps, or both, with each required table column and map feature specified. For example, the data requirement "Operational status of impacted transportation critical infrastructure" is linked to several key information requirements and is described both as a table with 9 data columns per impacted infrastructure node and as a map with 3 map features per impacted infrastructure node. The custodial owner of the dataset serving as the source for the data requirement was also specified.

Table 73.3 Example data requirement and metadata attributes

Unique ID	Data requirement	Source	Data format
D6	Operational status of impacted transportation critical infrastructure	General transportation status data are collated in the DOT map (DOTMAP). Airport status data are sourced from the Federal Aviation Administration (FAA). Port status data are sourced from the USCG Homeport platform. Rail status data are sourced from DOT and private rail providers. During Black Sky events: Same, but likely to be delayed if relevant agencies' facilities lose continuity of operations from outages. Initial transportation infrastructure status reports may need to be exchanged via phone call. Custodial owner of data: DOT, FAA, USCG	Table:transportation node name, type (e.g., bridge, road, port), name, service area (state, county), address, lat/long, status (open, restricted, closed), estimated date and time of reopening, and timestamp Map:for each state, the location of each transportation node/road is mapped, with icon or line color indicating status (open, restricted, closed), and timestamp

Custodial owners are identified, including agencies at all levels of government and in all sectors, private organizations, and industry. For both data requirements and information requirements, the voice communications capacity needed to ensure the data can be collected and discussed if electronic messaging and file transfer are impossible is also specified (e.g. point-to-point, teleconference, live video). Several data requirements identified are dynamic, meaning they need to be refreshed regularly during Black Sky operations to be kept current and relevant. Others are static, meaning they do not need to be refreshed.

An example data requirement is provided in Table 73.3. The complete data requirements table is provided in Appendix C.

73.3.3 Aligning Operational Missions with Information Requirements

A fully functional ECOM and BSX platform must successfully include, integrate, and synthesize an array of data requirements to support operational mission requirements. The systems-level framework characterizes complex interconnections and interdependencies between all three types of requirements aligned in the hierarchy. Fig. 73.3 illustrates these interdependencies for two critical operational mission requirements. For example, the data requirement "Active hazard

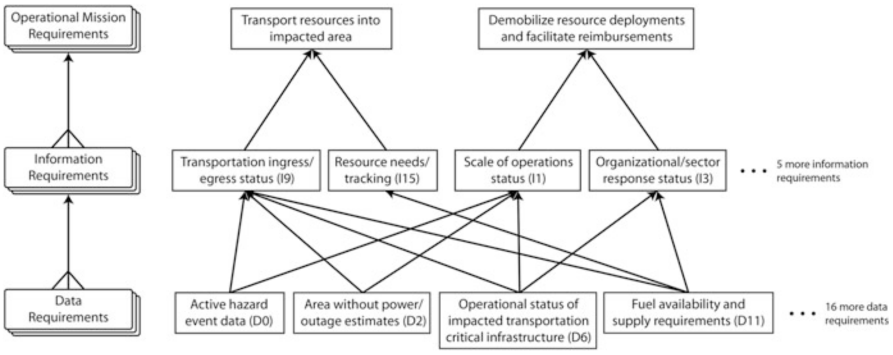

Operational Mission Requirements
Information Requirements
Data Requirements
Transport resources into impacted area
Demobilize resource deployments and facilitate reimbursements
Transportation ingress/ egress status (I9)
Resource needs/ tracking (I15)
Scale of operations status (I1)
Organizational/sector response status (I3)
... 5 more information requirements
Active hazard event data (D0)
Area without power/ outage estimates (D2)
Operational status of impacted transportation critical infrastructure (D6)
Fuel availability and supply requirements (D11)
... 16 more data requirements

Fig. 73.3 Alignment of systems-level requirements framework to examples. Not all supporting information requirements and data requirements are shown

event data" is needed to support both operational mission requirements, and both would be affected by any disruption to this data requirement, such as a failure to collect or update the data during Black Sky operations.

The complexity of interconnections seen even in the relatively small subset of examples in Fig. 73.3 suggests that a robust systems-level framework is needed to ensure efficient implementation of data requirements in the ECOM and BSX platform. The framework allows data requirements to be directly mapped to the operational mission requirements they support such that only data requirements necessary during Black Sky operations are included. Adopting such a systems-level framework in the implementation of an ECOM and BSX platform would ensure that all data required for operations are provided by the platform.

73.4 Conclusion

Here, we describe a systems-level ontological framework to list and characterize Black Sky requirements. Specific operational mission requirements, information requirements, and data requirements are defined within the framework, including the hierarchical relationships between them. Data requirements are characterized by their relative load requirements for information transfer in order to support future work in estimating the absolute disk space and bandwidth requirements for the ECOM and BSX platforms.

73.4.1 Incorporating Data into the Platform

In developing the ECOM and BSX platform, the specific data and information requirements defined here need to be incorporated to ensure that the necessary actionable information is available to the decision-makers and groups tasked with meeting operational mission requirements. The systems-level framework and database developed links specific data sources to operational mission requirements to ensure that data required to support the response and recovery would be built into an ECOM and BSX platform. This approach complements emergency response plans, which set forth agency- and organization-level emergency operations activities in prescripted mission assignments and other forms. The data and information requirements defined are highly specific to clarify the source and format of actionable information to drive response and recovery activities. The operational mission requirements are flexible, emphasizing the task that must be completed over the group that should be responsible for it and primarily serving to motivate the incorporation of specific data in the platform.

73.4.2 Providing Information in the Proper Form

As the ECOM and BSX platform is developed, it will need to be designed to incorporate the information requirements and formats required to support operational mission requirements. By tailoring the data and information provided to operational missions, the platform will be much more likely to meet the needs of the end users in the emergency management community. Likewise, the information requirements will need to be presented in a format that is immediately useful to the user community and clearly addresses specific decisions in response and recovery operations.

73.4.3 Importance of Adopting a Systems-Level Framework

The complex interdependencies between data requirements and operational mission requirements suggest that a systems-level framework should be used to guide the implementation of the ECOM and BSX framework, as was described here. Most information requirements rely on the successful inclusion, integration, and synthesis of not just one dataset but several to effectively support Black Sky operational mission requirements. This finding highlighted the utility of clearly articulating the specific data elements that comprise each data requirement and the connections between all system components from the data level to the operational action level.

73.4.4 Ensuring Access to Black Sky-Relevant Datasets

As the ECOM and BSX platform is developed and implemented, it will be critical to work with the custodial owners of data sources. Expectations of what data will be provided (as defined in the “Form” metadata attribute of the relational database) and when will need to be defined in advance to ensure optimal function of the platform. A diverse range of custodial owners for data sources to meet data requirements included federal agencies such as FEMA and DOE, state agencies, and private sector or industry owners. Establishing close relationships with these groups during platform development will ensure that data are available to the platform when needed during Black Sky operations; this coordination is especially important for dynamic datasets that must be continually updated during the event.

Once the ECOM and BSX platform is implemented, the included data requirements should be regularly updated and refreshed to ensure that newly developed or refreshed datasets are kept current. For example, static datasets describing the population with durable medical equipment are updated on a regular schedule during steady-state. The ECOM and BSX platforms must be loaded with or able to access the most recent versions of such datasets to ensure Black Sky operations are based on the most recent data. Similarly, plans to access updates to dynamic datasets during the event should be preestablished and tested.

73.4.5 Technical Specifications for the Platform

Additional work building on these results is required to determine the absolute data transfer load that the ECOM and BSX platform must be designed to accommodate, including choice of information technology solutions. This work lays the foundation for determining the technical specifications for the platform by defining the relative data transfer load for each data requirement (i.e. the number of and type of data fields needed per data requirement). However, these results do not address information technology and device considerations, such as the choice of operating system, geographic information system application, security software, and hardware setup, which will impact the absolute number of bytes that the system must handle. Once these considerations have been addressed, the results of this work can be used to directly support disk space and bandwidth requirement calculations. It will be essential to optimize requirements to ensure efficiency during a resource-limited Black Sky scenario.

The voice communications bandwidth requirements for the platform will also require careful consideration. In addition to supporting the use of data during Black Sky operations, the platform will be required to support robust voice communications between parties involved in the response and recovery. The relative voice communications bandwidth requirements associated with each data requirement and each information requirement have been defined here, but additional analysis will be required to estimate bandwidth requirements based on the expected frequency of point-to-point and teleconference calls, the number of concurrent calls expected, the average length of calls, and the data transfer per unit time required for each type of call.

Acknowledgments This work was funded by the EIS Council [13], a nonprofit research entity established by the electric utilities and their major customers, whose support is appreciated.

Note: Appendices can be found online at www.talusanalytics.com/CSER2017

References

1. Siegel N, Ferren B (2016) Emergency Communications System (ECOM), A Technical Report for the Electric Infrastructure Security Council, Aug. 2016
2. This paper is one of four that is part of a coordinated panel that is considering the topic of "Protecting Electric Power Sources". This panel was chaired by Neil Siegel
3. US Department of Homeland Security (DHS) Federal Emergency Management Agency (FEMA), 'Federal Interagency Operational Plans' (2016) [Online]. Available: https://www.fema.gov/federal-interagency-operational-plans. Accessed: 5 Jan 2017
4. "Data Requirements for ESF #12 Information Sharing: Interagency Results," US Department of Energy (DOE) Office of Electricity and Energy Reliability (OE), Interagency report, July 2015
5. Secretary Napolitano Unveils "Virtual USA" Information-Sharing Initiative (2009) [Online] Available: https://www.dhs.gov/news/2009/12/09/virtual-usa-information-sharing-initiative
6. DHS Science & Technology, Virtual USA Fact Sheet. [Online] Available: https://www.dhs.gov/sites/default/files/publications/Virtual%20USA_0.pdf
7. US Department of Homeland Security (DHS), 'National Response Framework, Third Edition' (2016) [Online]. Available: https://www.fema.gov/media-library/assets/documents/117791. Accessed: 5 Jan 2017
8. National Strategy for Information Sharing and Safeguarding (2016) [Online] Available: https://www.whitehouse.gov/sites/default/files/docs/2012sharingstrategy_1.pdf
9. Information Sharing Environment, Information Interoperability Framework (2014) [Online] Available: https://www.ise.gov/sites/default/files/FINAL%20-%20ISE_I2F_v0%205.pdf
10. Incident Management Information Sharing Sub-Committee, A Subcommittee of the National Security Council Staff Information Sharing and Access Interagency Policy Committee. [Online] Available: http://nisconsortium.org/wp-content/uploads/2014/06/About-the-IMIS-SC.pdf
11. US Department of Homeland Security (DHS) Federal Emergency Management Agency (FEMA) Modeling and Data Working Group (MDWG) (2015) "Systems Analysis of the Data and Models Used for Federal Emergency Management," Final report, Feb 2015
12. DHS SAFECOM Interoperability Continuum, A Tool For Improving Emergency Response Communications and Interoperability. [Online] Available: https://www.dhs.gov/sites/default/files/publications/interoperability_continuum_brochure_2.pdf
13. More information about the EIS Council Black Sky project can be found at www.eiscouncil.org
14. US Department of Homeland Security (DHS) Federal Emergency Management Agency (FEMA) Modeling and Data Working Group (MDWG) (2015) "Modeling and Data Working Group Flood Scenario Analysis," Draft report, Nov 2015
15. US Department of Homeland Security (DHS) Federal Emergency Management Agency (FEMA) Modeling and Data Working Group (MDWG) (2016) "Modeling and Data Working Group Biological Scenario Analysis," Draft report, Aug 2016
16. US Department of Homeland Security (DHS) Federal Emergency Management Agency (FEMA) Modeling and Data Working Group (MDWG) (2016) "Modeling and Data Working Group: Recovery Phase Analysis," Draft report, Aug 2016

Part X
Systems Engineering Education

Chapter 74
An Architecture Analysis of a Cyber Secondary School as a System of Systems

Cheryl Emerson and Tommer Ender

Abstract Education is important to prepare individuals to survive and contribute in contemporary society, and improving education can reap great rewards. There is not much published education research related to systems architecting in general, or for a cyber secondary school (CSS) in particular. Systems engineering processes, architecture considerations, and architecting models are discussed analyzing an architecture for a CSS as a system of systems (SOS). Architecting a CSS would make a valuable contribution to the field of systems architecting as a novel endeavor without prior publication as well as provide a model for experimenting with proposed changes as improvements to the school system. Furthermore, instead of testing for defects, the authors propose a systems engineering approach to architect for ultraquality, which represents a radical change to education system design and management.

Keywords System of systems • SOS • Systems architecture • Cyber school • Education • Online school • Secondary school

74.1 Introduction

Educational systems and compulsory education have existed since long before the Common Era, so school systems and education are not unprecedented. Online education is the newest iteration of educational systems. Instead of being revolutionary, or even evolutionary, online education is generally what has always been done, using software and the internet for distributed delivery, instead of being co-located in a physical classroom. What is unprecedented is the establishment of online schools (primary and secondary) independent of physical school districts. Neither a private school nor a state school in Georgia, the online school exists as a for-profit, hybrid charter school, not bounded by the physical constraints of a local

C. Emerson (✉)
Georgia Tech PMASE, Atlanta, GA, USA
e-mail: cemerson30@gatech.edu

T. Ender
Georgia Tech Research Institute, Atlanta, GA, USA
e-mail: Tommer.Ender@gtri.gatech.edu

A.M. Madni et al. (eds.), *Disciplinary Convergence in Systems Engineering Research*, https://doi.org/10.1007/978-3-319-62217-0_74

school district, encompassing the entire state, as well as some students beyond state boundaries. The online school that is the system of systems for students, their families, educators, and society, is unprecedented.

The students, their families, and educators have a clear vested interest in the education system, and may be considered the primary stakeholders in this effort. Society at large, as well as the government specifically, are also stakeholders, but in a more general sense, as they are impacted by the outcome, but not necessarily the ongoing processes of education. Additionally, for the cyber school being described, the corporation is the owner stakeholder and typifies that the payers of the system are not the users of the system. The corporation funds the curriculum, educator salaries, and infrastructure, and the government pays the corporation to do that. Students and their families do not pay, but are working toward completing and earning credits toward graduation. Educators are employed by the corporation to present the curriculum and support students' progress.

The purpose of any education is to prepare individuals for future performance (survival), as well as being an agency for social change. In addition to the basic abilities of reading, writing, and computing, students learn how to learn, where to find information, how to problem solve and think critically, behave appropriately, and work together. Educational systems not only provide curriculum and instruction for learners, but also document and certify students' accomplishments. The specific purpose of the online high school is to provide the standards-based curriculum, along with services to prepare students to complete state testing, provide the testing, and award credits as specified by the state, in order to complete graduation requirements. The purpose of the cyber secondary school (CSS) has been identified at the outset, but the architecture continues to evolve haphazardly, without the structure and support an architecture would provide.

The CSS is a sociotechnical system of systems enterprise. Every individual associated with the school, being a human, represents a system of systems and display what Bjelkemyr et al. [1] list as the characteristics of a system of systems: evolutionary behavior, self-organization, heterogeneity, emergent behavior, and a network. The school has qualities, which according to Maier [2] typify a system of systems (SOS), such as geographic distribution, and operational and managerial independence of systems within the system. For example, students and teachers are located within their own offices, which they control, geographically distributed across an entire state and remote from both the administrative offices located in a city within the state, and the corporate offices located in another state entirely. Additionally, according to Maier, the purpose of a SOS would be expected to be fulfilled by the emergent behavior of all the systems working together and not attributable to any component system. Secondary school students cannot graduate without a school, and the school system cannot exist without students; any individual can of course educate him or herself independently, but then is neither tested nor certified in accordance with the state. A school may be considered a system which produces "educated" individuals, and the combined behavior of all of the constituent systems within the CSS SOS is required in order to achieve the principal purpose of the CSS.

The CSS represents a hybrid type of managerial control. Students and families represent SOS that presents what is described by the Department of Defense (DoD)

[3] as acknowledged managerial control. Changes in the system are based on collaboration; the student systems are independent, possessing their own objectives, management, resources, and processes, which may or may not be in support of the school-wide SOS. The manner in which they operate and behave is not subordinated to the management of the school system. Changes to the student systems must be based on agreement and collaboration, not top-down authority from the SOS manager, or in this case the head of school. As with directed managerial control, SOS objectives, management, and funding exist without any authority over the constituent student systems. The integrated SOS was built and is managed to fulfill specific purposes, as, for example, with staffing and software. In this regard, it is under directed managerial control as defined by the DOD [3], which describes the relationship systems (teachers) have with the school SOS, centrally managed with the expectation that the corporation determines the purpose. Constituent systems (teachers) maintain operational independence, but normal operation is subordinate to central management.

An architecture analysis for a CSS benefits not only the system of observation by documenting and evaluating the system in pursuit of improvement but also society at large through the contributions of educationally improved individuals produced as a result of system improvements and contributes as well to the nascent field of SOS architecting with a novel exploration of a CSS as an SOS.

74.2 Literature Review

Education is the key to societal change, and Gillard [4] reported the key 17 global development goals essential to eliminate poverty and hunger and improve health; all can be accomplished by improving education. The benefits of improving educational systems are undeniable. The literature search for articles relating to architecting, education, and SOS revealed a publication paucity, with nothing relating to architecting a cyber school as an SOS, although there is literature on physical architecture and the educational domain [5, 6].

Applying systems thinking in education was advocated by Betts [7] in 1992. The article potently identifies system structure and feedback loops as the genesis of unwanted (emergent) behavior of a system. He advises that the solution is to understand and change the structure. As early as 1999, Kaput et al. [8] presented the United States educational system as a complex system that could be improved and strengthened by the use of a complex systems approach and proposed the implications of alternative systems even before the advent of online education. Another article [9] on complex social systems and emergent behavior reveals that these kinds of problems can only be solved by analytical methods through the use of systems thinking, selecting the process to fit the problem, and understanding the system dynamics.

Eadie [10] described the importance of a designed structure and process for a school board in 2005, and while there are guidelines for website architecture for education [11] and user interface heuristics, [12] there is an absence of literature on the subjects of architecting, design, or systems analysis for online secondary schools. Higher education has become highly dependent on online education according to Burnette, [13] while Davis [14] posits that even though it exists in all 50 states, virtual education is still understudied and that the important questions to be asked are how best to implement online learning and what determines success.

74.3 Systems Engineering Processes

The traditional top-down/bottom-up systems engineering process is not directly applicable to architecting the CSS because the constituent systems are at different phases in the systems engineering process lifecycle. Without synchronicity of development across the constituent systems, the traditional system engineering approach does not work on the SOS as an entity. Additionally, the CSS is an enterprise, and according to Nightingale and Rhodes [15] enterprises do not lend themselves to traditional decomposition approach to complex systems because leadership and enabling process must be considered in parallel. The DoD [3] lists seven core elements to describe an SOS environment: Translating capability objectives, understanding systems and relationships, assessing performance, developing an architecture, monitoring and assessing change, addressing requirements and solution options, and orchestrating upgrades to the SOS. These elements, which Rebovich [16] describes as mega processes, provide context for applying systems engineering processes to systems of systems architecting.

Relationships between constituent systems are not documented for the CSS, either over time or otherwise. The organization structure, processes, and plans seem to be continually changing, both in terms of personnel and responsibilities, making it difficult to plan and maintain documentation. The focus should be on the contribution of constituent systems to the SOS capabilities and the relationships between them, rather than system level boundaries and interfaces. Synchronization across the SOS and between constituent systems must be planned. The applicable systems engineering processes to this SOS architecting megaprocess are technical, logical, and management analyses.

Modeling and simulation currently are not used in the CSS for architecting purposes. There are limited data modeling of student results to plan for reteaching at the course level, but there are neither inter- nor intrasystem models. Metrics and methods are the keys to assessing performance to desired capabilities, and creating models for simulation are important for technical process validation, technical management and assessment, as well as to be able to track risk mitigation and data flow and control.

In order for the CSS to be architected, an architectural description would need to be completed and the architecture developed. Concepts of operations, functions, relationships and dependencies, and data flow would be documented and modeled. There needs to be a technical framework to use to analyze and evaluate change and to provide for options and trades. The systems engineering process to be applied at this stage would be selecting design solutions, in addition to continued requirement development and logical analysis. Monitoring and assessing change is not a process that the CSS follows. The CSS does experience changes in technology and environment, but it is not monitored, nor assessed, and certainly not in the context of relationships between constituent systems, or planning for future changes. The environment is one of fire control rather than fire prevention, and the fires are not documented to look for patterns. Monitoring and assessing change is yet to be for the CSS.

Evaluating requirements and solution options is a megaprocess [16] that would address which SOS requirements an architect for the CSS should/could/would implement, then evaluate the options, to derive, decompose, and allocate those requirements to the specific constituent systems. The result would be a detailed implementation for the CSS SOS capability. In addition to continuing to develop requirements and design solutions, analyzing, planning, and managing engineering processes apply here. Once an architecture has been documented for the CSS, any changes or upgrades made to constituent systems must be evaluated to ensure the implementation supports the SOS. This is done by integration and development testing and evaluation, and this megaprocess involves planning and facilitating the testing. This would involve most of the systems engineering technical processes: implementing, integrating, verifying, validating, and transition.

74.4 Architecture Considerations

The architecture of a system is the structure of the components, connections, and constraints according to Maier and Rechtin [17]; in the case of the CSS, one of the components, the student system, can be considered the product of the system, as well as a constituent system. Given the unique nature of the topic, and the total lack of systems engineering approach, architecting would of necessity rely heavily on heuristics.

74.4.1 Relationship Between Constituent Systems

The constituent systems that comprise a CSS include students and their families, educators, and the communication and educational software that deliver instruction and the curriculum, as well as the business enterprise systems. The CSS is not strictly hierarchical, but rather represents a holarchy, or heterarchy in which the systems may be hierarchical but also share horizontal as well as vertical relationships. High school educators are generally organized into subject-specific content departments, and also grade level teams for homeroom and other nonacademic custodial duties. It is a collaborative system, in that component systems (systems) are managed by and for their own purposes, as opposed to the purpose of the CSS as a whole. The CSS is a heterarchy with horizontal, diagonal, and vertical relationships between systems. Daily instruction may see teachers interact with as many as several hundred students, while each student may interact with from no to several teachers per day. The exchange of information or requirements describe the information relationships for every SOS. The constituent systems of the CSS can operate independently, but would not then fulfill the mission of the CSS. They have operational independence, but the result of the conjoined operations is the product of the system, or "educated" student systems. There are no generation relations in which one constituent system replaces another in the CSS. One might consider the replacement of one school system by the next level constituent system as a generation relation, but that would apply to the CSS environment, not the CSS.

74.4.2 Specific Architecture Considerations

According to Cole [18], specific architecture considerations, which warrant attention for a SOS, are: autonomy, complexity, diversity, integration strategy, data architecture, and system protection. Certainly there is technical and operational autonomy for each system within the CSS SOS. Each student and teacher represents an independently located system within the system and have separate and different infrastructures, operating budgets, and management, with variable organizational relationships to other enterprises. The complexity in the system is in the interrelationships between the systems and the dynamic nature of the enterprise. The goal is to minimize the complexity by minimizing the interrelationships. It is natural for constituent systems to specialize and be optimized to perform their primary function, for example, subject area teachers are specialized to a content, or some teachers are better with gifted learners, while others are more successful with populations of English language learners, as it is difficult to be all things to all people all the time. Multiple systems in an SOS will perform similar or even identical functions and often do so in very different ways.

SOS designs should be considered for diversity, eliminating one-size-fits-all thinking to reduce common node failures. Certainly the student constituent systems are diverse in geography, ability, interest, intellect, and available resources. The diversity of needs must also be considered; not all students are created equal, and so represent constituent systems with diverse needs. Additionally, different systems are also motivated by needs that may change over time. The business case also might change as well. Constituent systems evolve differently based on different environmental pressures, strain on the SOS must be accounted for in architecture of SOS and this is true too of the CSS. No two teachers and students systems are identical.

The CSS presents a poor example of integration strategy and has a problem with communication between platforms. The lack of an integration strategy for the CSS is an apparent deficit in organization alignment and system to system interfaces within the context of the SOS. The bridging strategy employed by the CSS is human in the loop to decode from one system and recode into another. Data architecture concerns arise when posting shared data for use by more than a single system. It is important to coordinate any necessary changes in structure or semantics. For the CSS SOS, there are multiple student information management systems, and they do not communicate, which means that humans in the loop have to make double entries. Unfortunately, with double the work comes more than double the potential errors. Preventing unauthorized access, identifying authorized users, and restricting modifications to those with authority are critical concerns for system protection. The CSS is accountable to Family Educational Rights and Privacy Act (FERPA) [19] privacy guidelines, in addition to limiting use to authorized-only users through password protections and IT security. Guaranteeing the identities of resource consumers and providers is another deficit of the CSS, as students will log in using learning coach (parent) accounts, and learning coaches as well as others may also log in as students.

Critical elements are those elements within a system that a system could not exist without and is true for constituent systems within a SOS. In the case of the CSS, systems known as students are critical elements, without which there would be no SOS or school. The same is true of teachers for any school, and for equipment, software, and communication systems, including the internet, in particular for an online school. Data, or records and student accountability systems also represent elements critical to the CSS, without which the SOS could not exist. Architects should be concerned primarily with the interactions between constituent systems of a SOS, and intrasystem level decisions are best left to the subject matter experts. So it might be argued that instruction decisions are the purview of teachers. Certainly no architect would direct what or how code is written for a particular platform, beyond providing expectations and requirements for the intersystem interactions, and perhaps schedule oversight, as, for example, for a supply chain. But the closest thing in the system to an acting architect is the head of school, who is not trained in architecting and occasionally focuses beyond intersystem interactions to intrasystem choices. Nonetheless, the head of school is a critical element as one might argue that without a head, the body is lifeless.

The critical factors for architecting SOS solutions to succeed according to Cole [18] are: robust design, architecture alignment, governance, and description. The CSS does not have an architectural description nor a model, let alone a well-designed one. A robust design is insensitive to changes. Business case and schedule robustness are key aspects that drive policy for a robust design. A robust design ensures that the SOS serves the intended purpose under a full range of environmental conditions. Systems reliability can be improved by avoiding failure modes as a strategy during design conceptualization. Allowing contingencies for delays is necessary. Some of the CSS constituent systems (students) will not be robust with regard to schedule, so the SOS as whole must have contingencies for delays preplanned. Being able to accept and accommodate technology changes is another key aspect to robust design. If a planned improvement within the SOS is to a critical capability, it is more robust to have a contingency approach for critical needs which do not measure up, particularly with introduction of new technology. Within the

CSS SOS there exists a variety of hardware, capabilities and software versions that protocols would need to accommodate, for example browser support, as well as language, and assistive technology.

Similarly to robust design, architecture alignment has three main aspects to consider, organizational alignment, business process alignment, and technological alignment. These refer to aligning organizations, business processes, and technologies to operate in a system of systems framework. The underlying infrastructure and architecture already exist for the CSS and so must be accounted for and/or modified. Governance is a term for having each system within an SOS operate under the same general rules and framework. Architecture governance is decomposed into the role of a system within the SOS, and the interfaces of different systems within an SOS. Successful architecting requires that a system within the SOS be able to make changes in a controlled and coordinated manner, taking into how on constituent systems might be affected. The final element of SOS success is a well-defined architecture description. This requires depictions of the SOS from various viewpoints, as no single view is cohesive enough to describe an entire system. Cole [18] points out that the once the frame work is complete, it serves both as a model and the roadmap to follow.

74.4.3 Architecting Models

A list of what the client wants and the feasibility of the system to fulfill the desired purpose(s) describes a purpose model. According to Maier and Rechtin, [17] in the usual manner modeling would begin with objectives or requirements, and then the modeling method and language would emerge. As this analysis is of an extant system that is the result of unguided evolution, one must work backwards, in effect reverse engineering from the system as it exists to describe the architecture. The list of client wants, according to the charter, [20] includes directions for the system to:

- Provide an educational environment for all students
- Include differently-abled students
- Comply with state and federal department of education requirements (ESSA, ESEA, IDEA, FERPA)
- Provide a virtual, child-centered online environment for students to construct their own learning experience
- Be economically sustainable
- Use generally accepted accounting practices
- Meet financial reporting deadlines
- Promote positive school experience
- Engage students, parents, and teachers

From the perspective of the SOS corporate owner, the SOS must be profitable. These, together with other requirements, would generate a SysML requirement diagram which could be verified through a requirements satisfaction diagram. Ideally the constituent systems purposes would be the same as, or a subset of, the SOS purpose. In the case of the CSS, this is generally the case.

To certify a system, the functionality that the client wants is assessed so that the client can be sure they are getting what they have paid for; a system that is both useable and useful. Certifying a CSS architecture presents an interesting conundrum. Ostensibly, a certified system would produce ultraquality educated students at a profit for the corporation. Charter schools in the state are "certified" according to their charter, and therefore must demonstrate proficiency in terms of attendance, pass rates, and graduation rates for all students, including demographic subgroups. There is no certification for the CSS system of systems architecture, because it has not yet been developed. In order for schools to be accredited, or keep their certification, they must have certified teachers, and the number of students who attend, graduate, and complete and pass state testing, is tracked. In this case the most likely client would be the corporate owner of the online school, and the "tests" of the SOS, or improvements to the architecture, would be improvement compared to past history with increases in ratings based on financial efficiency and school climate, lower turnover of students and staff, with better passing and graduation rates for students.

The elements and interfaces of the system, or representations of them in accordance with standards, laws, or policies, constitute a form model. Making a physical scale model would not be an appropriate way to model the CSS, as it exists as a distributed, virtual system. A model of the CSS can be generated using SysML block definition and internal block diagrams, and the CSS environment is presented in Fig. 74.1. The pink student and teacher blocks are critical elements without which the SOS would not exist and represent the enrollment and academic service constituent SOS, respectively. The blue colored blocks identify systems, and the pink and purple blocks label systems of systems.

Behaviors, or what the system does, are described by use cases. The CSS use case is presented in Fig. 74.2. The highest level use case of the SOS being described is society as the actor using the system to educate students, or in other words, prepare students for survival in contemporary culture. The system delivers curriculum and instruction to students within the CSS and certifies to society that the students have completed instruction via performance and time in course (attendance) and mastered the concepts or curriculum as evidenced by test or performance. The behavior of the system to provide instruction includes activities such as determining what the requirements are, and certifying that the requirements are met, either by testing or observation. The system also maintains and provides records for the society to use documenting what the student has been certified as having completed and accomplished.

74.5 Recommendations

Developing an architectural description for the CSS is recommended to be able to improve the SOS by applying what has been learned from architecting other enterprise sociotechnical systems and to architect the educational system for ultraquality instead of testing for defects. Establishing the baseline and then using models for comparison and evaluation of potential changes would be far superior to

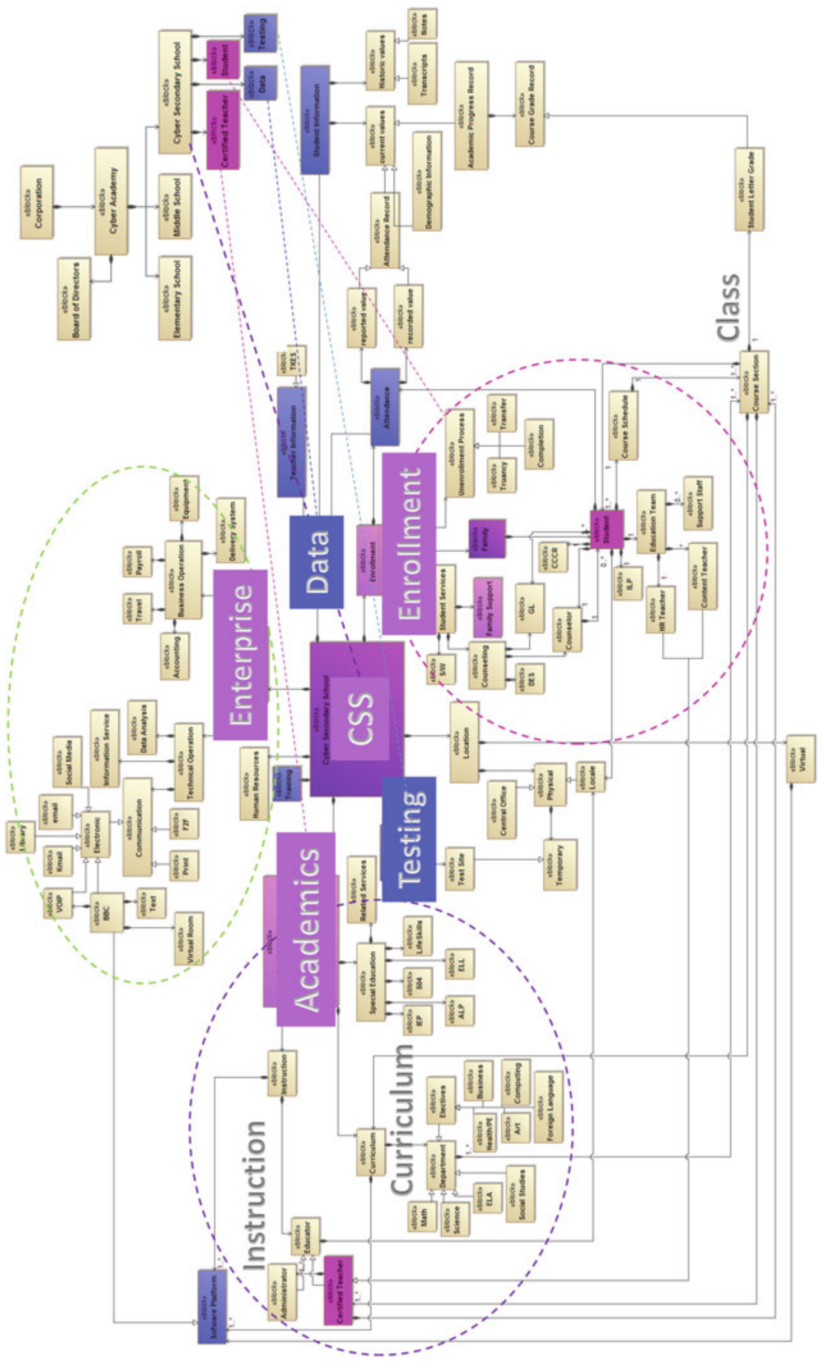


Fig. 74.1 CSS environment

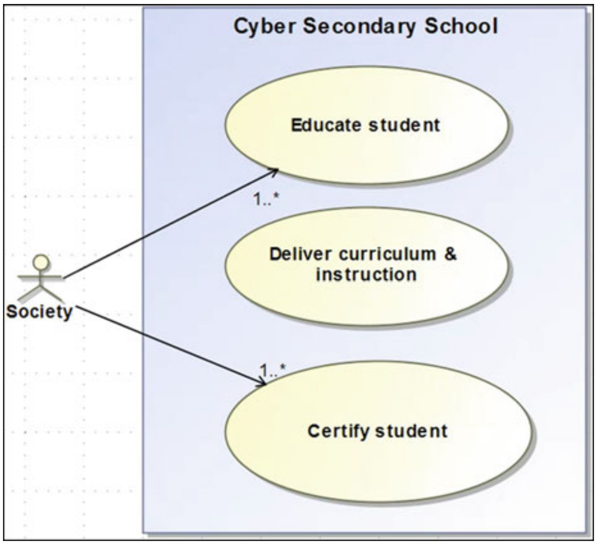


Fig. 74.2 CSS use case

the guess and check method currently in use. The evolved education system does have an architecture; it does not have an architecture description, but one can be created through a reverse engineering effort, generating a collection of products (artifacts) with which to document the architecture. The first step would be to construct the architectural description of the CSS SOS. Once the architectural description is complete, it must be evaluated for quality, which is the capability to meet the needs and concerns of stakeholders, emphasizing understandability, consistency, completeness, and analyzability, as well as feasibility, efficiency, and reliability of the SOS, and documented in accordance with ISO 402010 [21] recommended practice.

74.6 Conclusion

Education is the fundamental building block of society; it is the key to everything. Technology has evolved allowing for distributed delivery of education to students without a unifying physical locale. Not unexpectedly, behaviors and unintended consequences have emerged as the CSS has evolved without benefit of design, using the same approach for organization and operation that have been used for traditional nondistributed schools. The CSS, like any other distributed, complex system, must be architected. Although an architectural description for the CSS does not exist, this does not mean it cannot be done. By documenting the architecture of the CSS SOS, a baseline can be established upon which to improve. The models generated could then be used to evaluate changes in the physical system for the express purpose of improving system performance according to desired metrics, including student and teacher outcomes, by architecting for ultraquality.

References

1. Bjelkemyr M, Semere D, Lindberg B (2009) Definition, classification, and methodological issues of system of systems. In: Jamshidi M (ed) System of systems engineering principles and Applications. CRC Press Taylor & Francis Group, Boca Raton, pp 191–206
2. Maier M. Architecting principles for systems-of-systems. In: INFOED. Available http://www.infoed.com/Open/PAPERS/systems.htm. Accessed 21 Mar 2005
3. Office of the Deputy Under Secretary of Defense for Acquisition and Technology, Systems and Software Engineering. Systems Engineering Guide for Systems of Systems, Version 1.0. Washington, DC: ODUSD(A&T)SSE, 2008
4. Shapiro J. Education is the key to all global development goals (Q&A with Julia Gillard) Forbes Tech 10/03/2015. Available http://onforb.es/1PVVmvt. Accessed 13 Feb 2016
5. Earthman G, Lemasters L (2013) School maintenance and renovation administrator policies, practices, & economics, 2nd Expanded edn. DEStech Publications, Inc.
6. Ariani M, Mirdad F (2016) The effect of school design on student performance. Int Educ Stud 9(1). Available http://files.eric.ed.gov/fulltext/EJ1086689.pdf. Accessed 15 Jan 2016
7. Betts F. How systems thinking applies to education. Improving School Quality. Pages 38–1 educational leadership/no92/vol50/num03. Accessed 11 Jan 2016
8. Kaput J et al (1999) Planning documents for a national initiative on complex systems in K-16 education. Planning meeting June 18–20. Available http://www.necsi.edu/events/cxedk16/cxedk16.html. Accessed 11 Jan 2016
9. Systems thinking webpage thwink.org. Available http://www.thwink.org/sustain/glossary/SystemsThinking.htm. Accessed 11 Jan 2016
10. Eadie D (2009) Board climate matters. Am Sch Board J. Available https://www.nsba.org/board-climate-matters. Accessed 2 Feb 2016
11. The public facing advocate: in 8 simple steps. Available https://www.nsba.org/public-facing-public-advocate-8-simple-steps. Accessed 2 Feb 2016
12. Nelson J. Ten usability heuristics. Available http://www.useit.com/papers/heuristic/heuristic_list.html. Accessed 11 Jan 2016
13. Burnette D (2016) Negotiating the mine field. Q Rev Distance Educ 16(3):13–25. Academic search complete, EBSCOhost. Accessed 28 Jan 2016
14. Davis M. A thirst for research. Available http://www.edweek.org/dd/articles/2012/02/08/02research.h05.html
15. Nightingale D, Rhodes D (2004) Enterprise systems architecting: emerging art and science within engineering systems. In: MIT engineering systems symposium, March 2004. Available https://esd.mit.edu/symposium/pdfs/papers/nightingale.pdf. Accessed 14 Feb 2016
16. Rebovich G (2009) Enterprise system of systems. In: Jamshidi M (ed) System of systems engineering principles and Applications. CRC Press Taylor & Francis Group, Boca Raton, pp 165–190
17. Maier M, Rechtin E (2009) The art of systems architecting, 3rd edn. CRC Press Taylor & Francis Group, Boca Raton
18. Cole R (2009) SOS architecture. In: Jamshidi M (ed) System of systems engineering principles and Applications. CRC Press Taylor & Francis Group, Boca Raton, pp 191–206
19. Family Educational Rights and Privacy Act (FERPA). Authority: 20 U.S.C. § 1232g; 34 CFR Part 99. Available http://www.ecfr.gov/cgi-bin/text-idx?c=ecfr&SID=16796a773ac48f980cdfaed80b1fa94a&rgn=div5&view=text&node=34:1.1.1.1.33&idno=34. Accessed 8 June 2016
20. Georgia Cyber Academy Charter State Charter Schools Commission (2013) Available https://scsc.georgia.gov/sites/scsc.georgia.gov/files/related_files/document/GeorgiaCyberAcademy_FE.pdf. Accessed 28 Jan 2016
21. ISO/IEC/IEEE 42010:2007(E) Systems and software engineering – recommended practice for architectural description of software-intensive systems. Available https://files.t-square.gatech.edu/access/content/group/gtc-6223-db57-51b9-98d7-78fde0ed9d46/06.%20References_Readings/6.1%20Supplemental%20Readings/03.%20IEEE-1471-2000.pdf. Accessed 15 Jan 2016

Chapter 75
Systems Engineering: Making People Talk!

Cecilia Haskins and Kristin S. Ruud

Abstract In Scandinavia, a popular brand of pastilles uses the slogan "Läkerol makes people talk!" This phrase is the first that came to mind when considering a title for this report on the use of systems engineering approaches to organize and implement a master's program thesis. A valorized systems engineering process, SPADE, is used to design the research approach, and two systems thinking/engineering methods, the systemigram and swimlane diagrams, are used to conduct and document the research. The significance of this project was the way in which these relatively simple visualizations were able to engage the case company managers in the elicitation process and to facilitate an environment of interdepartmental cooperation. As the point of contact put it, "This was the first time they created a truly end-to-end view of their company purchasing, manufacturing and warehousing processes."

Keywords Elicitation methods • Systemigram • Swimlane diagrams

75.1 Introduction

Documented applications of systems engineering are less than 100 years old, but there is no scarcity of methods and approaches to support the systems engineer in their work [1, 2]. The case company is a producer of high-tech sensors and other systems, headquartered in Norway with offices around the world, including the USA, Spain, the UK, Australia, and Canada. In addition to manufacturing these products, the company offers different support functions, repair, and installation services. Key market sectors are offshore shipyard industries, naval, and research organizations. In these competitive markets, it is necessary to uphold a high service level as a strategic instrument to gain and keep customers. Thus, delivery precision is one of their KPIs and software is used to track flows in and out of the warehouses. Much of this paper is excerpted from the student's master's thesis submitted in June 2016 [27].

C. Haskins (✉) • K.S. Ruud
NTNU – Norwegian University of Science and Technology, Department of Mechanical and Industrial Engineering, Trondheim, Norway
e-mail: cecilia.haskins@ntnu.no; kristisr@stud.ntnu.no

A.M. Madni et al. (eds.), *Disciplinary Convergence in Systems Engineering Research*, https://doi.org/10.1007/978-3-319-62217-0_75

75.2 Research Method

In the university preparation courses, students are introduced to many research methods. When the time comes to write their thesis, they must chose approaches that are best suited to their research objectives. Systems engineering as a research method offers a systemic and systematic way to collect, analyze, and share the results of their research. Systems thinking is attractive as it seeks to understand the big picture, by recognizing that a system is composed of parts and their interconnections and that the way they are structured generates the system behavior [3]. Systems thinkers recognize the importance of changing perspectives and observing patterns in changes and trends [4]. The systems engineering perspective builds on systems thinking [5]. It is a structured discovery process [6] that, through an interdisciplinary approach, aims at enabling the realization of successful systems. In this research, the researcher chose to utilize the SPADE methodology.

75.2.1 SPADE: A Systems Engineering Methodology

The SPADE methodology was developed at NTNU as a systems engineering methodology for small research projects [7]. The acronym stands for stakeholders, problem formulation, alternatives and analysis, decision-making, and evaluation. The methodology is a valorized version of more complicated and robust systems engineering practices. As shown in Fig. 75.1, the methodology is represented visually as a circular model to emphasize the highly iterative nature of problem solving and to allow the user to enter the methodology at any point and traverse left, right, or across the diagonal. It is important to note that all activities are interconnected, thus a change or decision in one activity has the potential to influence other activities. However, common sense should suggest that selecting a solution before knowing the stakeholder requirements is not the best way to achieve success. SPADE has been applied numerous times and is described in previous articles [8–10].

Evaluation appears in the center of the framework because this activity touches all other activities. The process of evaluation is a control function to ask, *are we there yet?* It is a continuous question that supports flexibility and change.

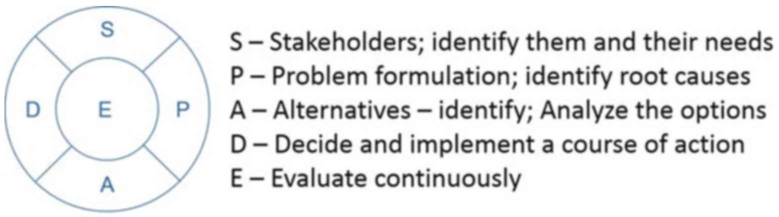

S
D
E
P
A
S – Stakeholders; identify them and their needs
P – Problem formulation; identify root causes
A – Alternatives – identify; Analyze the options
D – Decide and implement a course of action
E – Evaluate continuously

Fig. 75.1 SPADE methodology

Continuous evaluations facilitate introduction of new stakeholders and their late-arriving viewpoints to influence the scope of the problem. These new arrivals may change or alter the problem formulation and the relevant alternatives that are identified and analyzed. Based on this evolving environment, decisions should be reevaluated based on the measures of performance and success criteria established in the problem formulation stage.

This circular movement continues as long as the stakeholders are involved. Feedback and opinions from them are continuously evaluated. When stakeholders adopt new solutions or adjust their culture or the processes, feedback helps the researcher to update the current understanding of the situation, the problem formulation, the alternatives and the suggested solution. In this way, the suggested solution should best fit the problem and the stakeholder needs.

In this research, continuous evaluation has been supported by inputs from the literature study, and new knowledge has been used to reevaluate previously collected information, decisions, and understandings. The continuous nature of this research is also exemplified in the minutes of the meeting [27] as each face-to-face interaction in the case company involved more people who were engaged and happy to be heard and contribute to the research. This was very helpful when validating initial understandings about the internal processes of the company.

As a communication and a decision-making tool, systemigrams and swimlane diagrams were utilized to visualize company processes during the case work [11]. The following sections present these systems thinking methods in more detail.

75.2.2 Systemigrams: A Systems Thinking Method

Systemic diagrams or systemigrams are a schematic network [12] developed by John Boardman [13]. The network is composed of entities represented as nodes with arrows representing the interrelationships, as seen in Appendix A. Systemigrams are useful to capture the essence of complex systems. The aim of using the diagram in this research was to prepare for communication in meetings and interviews. As an artifact, they contributed to generating a shared understanding across the organization and a document that could capture this agreement.

The decision to use systemigrams in this research was based on the benefits provided by the graphical method. Systemigrams are a systems thinking modeling technique [14] that can visualize enterprise flow, inputs, outputs, beginnings, and ends. They create artifacts that are easier to understand and remember compared to a written text [12]. Moreover, as the graphical model allows for text emphasizing the relationships between the nodes, it is a more precise modeling tool when communicating key interactions between stakeholders. Ramsay et al. [12] argue that systemigrams will enhance an organization's ability for internal evaluation, and assist learning and facilitate continuous improvement of its processes [15]. Having visually expressed the system architecture, stakeholders are able to see the conceptual system and how interrelationships contribute to a complex system

[14]. One property of the systemigrams is the modeler's option to decompose the model into scenes that describe a story of what the systemigram represents whether a message or a process [16]. This is termed storyboarding, because a complete systemigram is not the end of a story, but possesses the basis for telling a story in a variety of ways. The researcher used storyboarding when communicating with stakeholders.

75.2.3 Swimlane Diagram: A Systems Thinking Method

A swimlane diagram is a graphical network that aims at visualizing responsibilities for certain activities defined to complete a specific operation or process [17, 18]. It is also referred to as an activity diagram by Object Management Group (OMG), or a cross-functional flow diagram by Microsoft [19]. The swimlane notation divides process models graphically in horizontal columns or vertical rows. At the beginning of each column or row, labels are used to name the partition. These partitions typically indicate departments, people, or other organizational units. The operation workflow is mapped out with details concerning activities, including decision paths necessary to progress the workflow to complete the operation. This includes showing the sequence of actions, identifying activity inputs, outputs, and the relationships between the activities. Responsibility is graphically allocated to organizational units by locating activities in specific swimlanes. Typically, key shapes are used to separate different categories of activities [19]. These key shapes visualize physical activities and triggering decisions activities. Moreover, shapes can be used to identify information that is stored in a digital database and on a written document.

The researcher chose to use swimlane diagrams in this research to visualize complex operations within the case company. Obtaining a holistic view of the case company included being able to effectively describe interactions between organizational units and activities and their allocated responsibilities.

75.2.4 Case Study

Triangulation is an approach where multiple sources of information are used to reinforce the evidence collected [20–22]. This is one approach to secure the quality of a case study. In this research, case data was collected through company documents and working papers, semi-structured interviews, and workshop sessions with key informants.

The research project revolves around a single company, which allows a researcher to acquire in-depth knowledge regarding the chosen company operations. The disadvantage of only one case is the inability to draw a general conclusion [22]. This is due to the individual characteristics of each case, available data, and the uncertainty of the interpretations done by the researcher. However, Meadows [23] states in her article that complex systems are complex and that it is dangerous to generalize them. By studying these complex systems, theories and preliminary findings can be further investigated and confirmed, patterns can be found, and theories can become more precise [24]. Experience regarding problem causes and symptoms can be developed and used as preliminary findings for further research activity for other complex case studies. Thus, this research sits in a continuum of departmental research projects.

75.2.4.1 Company Data and Working Papers

A thorough study of the case company's procedures was conducted by reading working papers and company documents. These were made available to the researcher by company representatives and through the project database. Reading such documents enhanced the researchers understanding of current company practices and issues, in addition to preparing the researcher for company visits and further information collection. Studying company papers is one type of qualitative data that is considered to be subjective [21]. To ensure correct understanding and to verify statements and descriptions of particular interest, additional steps were taken during company visits.

75.2.4.2 Semi-structured Interviews

The researcher chose to collect additional information through semi-structured interviews. These interviews were done on site in the case company's headquarters. Before each visit, the researcher communicated the topic and purpose of the interviews to the company, who in turn arranged for prime interview participants to be available on the day of the visit. Interview guides were prepared in advance, thereby providing a certain structure to the interviews, in terms of relevant topics and important questions [25]. A semi-structured interview approach was chosen for the flexibility given to the researcher to decide how and when to ask questions, depending on the interview situation [26]. The interviews were in conducted in English and Norwegian to reduce misunderstandings. The interview guides were prepared in English.

75.2.4.3 Workshop Sessions

Two workshop sessions were conducted at the company headquarters. These meetings aimed at obtaining a shared understanding between the researcher and the case company participants. Each workshop focused on collecting data to support different research objectives [20]. During the first workshop, managers and representatives from almost all departments were present. Systemigrams that represented the researcher's best understanding derived from project documents was used kick-start the discussion. The second workshop involved managers and representatives from all departments. A swimlane diagram was prepared for the session and validated during follow-on interviews.

After each session, reports were sent to the participants to collect their feedback and additional comments and to identify any misunderstandings by the researcher. These reports contained not only written text, but also illustrations and visualizations created during the company visits.

75.3 Results

When students begin their masters' thesis research they are often engaged to provide assistance to existing research projects ongoing in the department. At the same time, this provides them with a real-world contact experience while focusing, or limiting, the field of interest. In this situation, the student entered a project for a company that was in a state of reorganizational flux while trying to maintain a status quo of production and meeting existing and new customer orders. Existing project documentation reflected this uncertainty and did not provide a firm basis upon which to begin the research. The student researcher was also taking an overview course on systems engineering and felt that the application of systems thinking methods could help provide this foundation, as described above. In addition to research questions related to the project assignment, a third research question (RQ) was asked and answered:

RQ3: How can combinations of systems engineering methods support the collection of information and communication in the case study?

75.3.1 Mapping of Procedures

Taking an enterprise view, the researcher created a mapping of the end-to-end operations related to the manufacturing operations of the case company. The final mapping in a swimlane diagram covered four pages of the thesis and is understandably company confidential. The diagram illustrates departments and their responsibilities in terms of executing activities. The graphical illustration includes details concerning the sequencing of operations and the AS-IS information and material flow. A small sample is given in Appendix A. Systemigrams were an important precursor to the creation of the swimlane diagrams, and an example of the warehouse systemigram is given in Appendix B.

75.3.2 Observations Based on Applying the Systems Methods

The student observations are divided into three sections: preparing for, conducting, and postworkshop documentation.

75.3.2.1 Preparing for the Workshop Sessions

Systemigrams. The researcher used systemigrams before each workshop session. The systemigrams proved helpful in identifying missing details in the overall information available and understanding. Drawing systemigrams was hard work as the researcher first needed to learn the process of creating them and then did not have a thorough understanding of the systems at the company when trying to use them. However, the exercise helped the researcher prepare for the workshop sessions, provided some inspiration, and provided a foundation for determining what information was needed and how to retrieve it. In addition, mapping out the theme of the day helped narrow the scope of the workshop set achievable objectives for each session. Systemigrams were used in both of the two workshop sessions.

Swimlane Diagrams. Before the second workshop session, a swimlane diagram was drawn, presenting a detailed end-to-end view of the information collected up to that point. This diagram was based on a systemigram, but due to the qualities of the swimlane diagram more details became visible. The researcher decomposed operations into more detailed activities, expressing relationships and flow of communication. This schematic method was utilized to illustrate missing details in current understanding and uncover key topics for interviews and discussion. Similar to systemigrams, drawing swimlane diagrams is work intensive. However, advantages such as enhancing understanding and thorough preparation for the workshop session were observed as benefits by the researcher.

75.3.2.2 Conducting the Workshop Sessions

Systemigrams. During the workshop sessions the researcher observed that the systemigram proved to be effective communication methods. Even though this type of graphical network was new for all the workshop session participants, they quickly grasped the purpose and how to read the schematic network. The researcher observed that during the kick-off presentation, participants responded far better to the systemigrams, compared to written material or the researcher speaking alone. Key topics for interviews and discussion were introduced in the systemigrams, and in some cases discovered while drawing on a white board during the meeting. Participants became quickly involved, and were able to verify if the researcher's current understanding was correct. When incorrect or imprecise, corrections could be easily redrawn directly onto the systemigram.

During the first workshop session, the participants were able to draw an end-to-end systemigram of the information and material flow within the firm. The meeting became quite animated as managers commented on the work of the person at the board, or took the pen themselves and began adding to the diagram themselves. The project point of contact observed later that this was one of the most positive and collaborative meetings of this group and it set the tone for the remainder of the student research.

During the second workshop session, systemigrams were combined with semi-structured interviews. The researcher observed this combination to be a great success as valuable feedback and useful comments on current practices, challenges, and issues were communicated, and it was possible to obtain verification from participants not present when it was discussed.

Swimlane Diagrams. One aim of the swimlane diagram was to identify the extent to which software could support the company activities. The swimlane diagram was the basis for interviews during the second workshop, facilitating the workshop session participants to effectively identify missing details and to correct errors for the researcher. For each interview, participants were given a clean version of the swimlane diagram. This enabled the participants to draw directly in the diagram without making it too “messy.” Once again, swimlane diagrams engaged the participants, and the researcher has been informed that this visualization helped the participants understand their internal processes and relationships in a new way. The second workshop session included participants not involved in the first workshop. Having the swimlane diagram made it easy for the researcher to communicate the current state of collected data and to relate completely new information with the old.

Interestingly, visualizing previously collected information (the collectively drawn systemigram) in a new way, made participants respond differently (than the first time) and gave information from another perspective. In addition, the information became more precise, contributing to an increased holistic understanding of the situation. Visualizing who is responsible for a given activity, and how that activity is related to other activities, contributed to this outcome. Participants had an obvious ownership of their own tasks, and the preciseness of the swimlane diagram managed to verify current understanding and to extract more detailed information and meanings.

75.3.2.3 Postworkshop Documentation

Revising and updating the systemigrams and the swimlane diagram after the workshop sessions contributed to enhance the researcher’s holistic understanding and facilitated the study of company challenges. More importantly, working with these mapping techniques allowed the researcher to think about possible solutions to suggest alternatives for addressing the project objectives.

75.4 Discussions and Conclusions

This section contains the student researcher's observations to addresses RQ 3. Understanding the system architecture has been a vital aspect in this research for understanding the systemic issues of the production-inventory system of the case company. During the case study, the researcher addressed several stakeholders with current understandings and interpretations regarding diverse concepts. In the initiating phase of collecting information, systemigrams were an effective communication method as it supported precise communication. The literature supports the observation that systemigrams enable stakeholders to explore a range of perspectives while maintaining a single objective [14]. Perhaps what was less expected was the degree of engagement and commitment to 'get the picture right' that developed in the meetings between the stakeholders and the researcher.

Blair et al. [16] observed in his research that to preserve the readability of the systemigrams, the ratio of nodes and arrows should be approximately 1.5. Boardman and Sauser [13] and Blair et al. [16] state that the systemigram is particularly suited to express the strategic intent. The researcher observed both of these phenomena. When the drawing became too large or detailing became too precise, the systemigram lost its some of its value as an internal evaluation method that could assist learning and facilitate continuous improvement. Academic researchers suggest using the storyboard technique or developing a hierarchical structure of systemigrams as ways of avoiding a reduction in readability [13, 14, 16]. The authors consider these suggestions as valid if communication is written where the reader can go back and forth between pages, studying the interactions. However, in a workshop setting, having several interconnected systemigrams in a hierarchical structure is awkward.

Once the researcher experienced the strength of the systemigram technique to be able to illustrate the big picture and the rough lines and company operations at a high level, she decided to combine the systemigram method with the swimlane diagram technique to facilitate communication on a more detailed level. In other words, the researcher took the operation concept described in the end-to-end systemigram and some additional information collected from the case study and translated the information into the swimlane diagram template. This allowed the researcher to keep the readability and preciseness experienced from the systemigram, while simultaneously obtaining additional detailed information on activities, relationships, and responsibility.

Combining these methods in a stepwise procedure allowed the researcher to focus on the tactical relationships before going deep into details concerning operational interrelationships. The researcher observed that the stakeholders were able to provide information on a more precise level after the swimlane was established, indicating areas of responsibility. Figure 75.2 illustrates the synergy between these systems thinking methods in support of research. Additional observations made by the researcher concerning the combined use of systemigrams and swimlane diagrams are as follows:

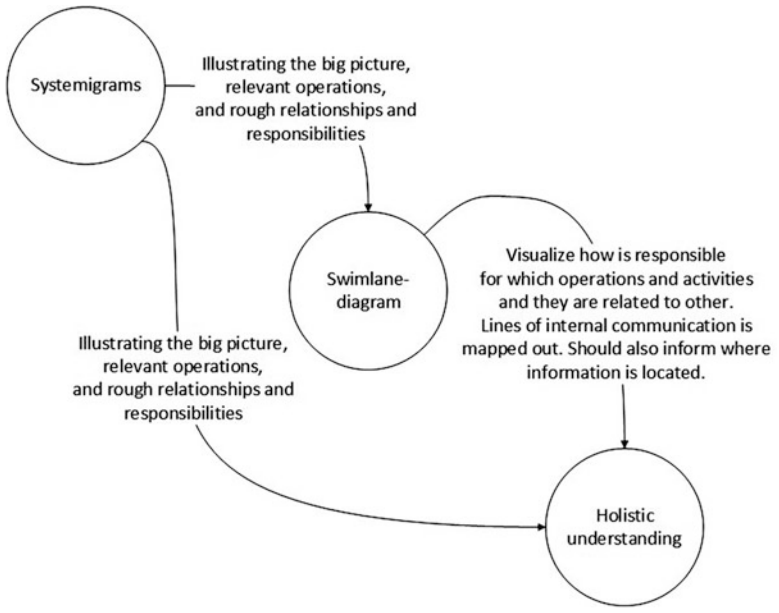

Systemigrams
Illustrating the big picture, relevant operations, and rough relationships and responsibilities
Swimlane-diagram
Visualize how is responsible for which operations and activities and they are related to other. Lines of internal communication is mapped out. Should also inform where information is located.
Illustrating the big picture, relevant operations, and rough relationships and responsibilities
Holistic understanding

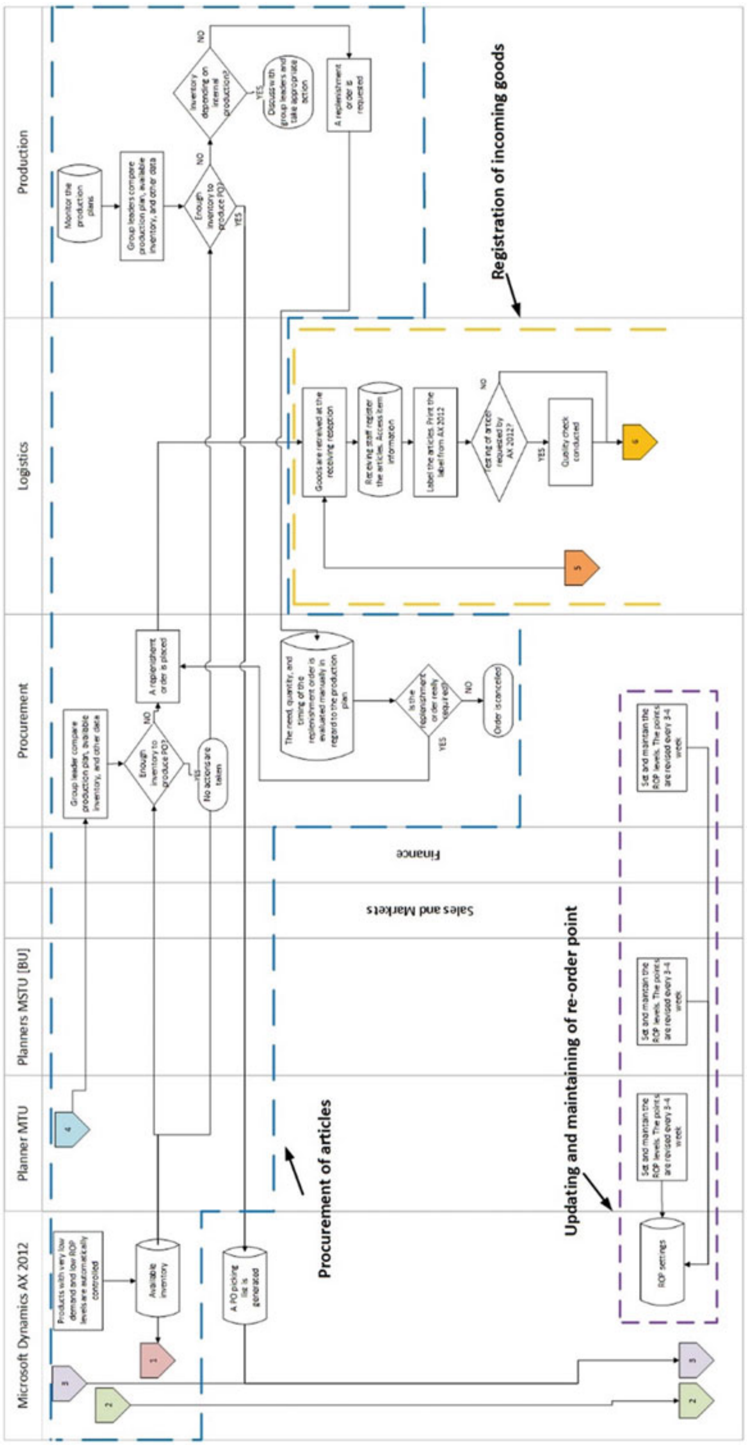
Fig. 75.2 Systemigram illustrating the synergy between systemigrams and swimlane diagrams

- Easy to understand
- Easy to learn
- Engaging conversation
- Involve workshop participants
- Enhanced understanding of current practices across the company
- Enhanced understanding of relationships

Acknowledgments The authors are grateful to NTNU and the case company for their cooperation in this research and to the Norwegian Research Council for their funding and support.

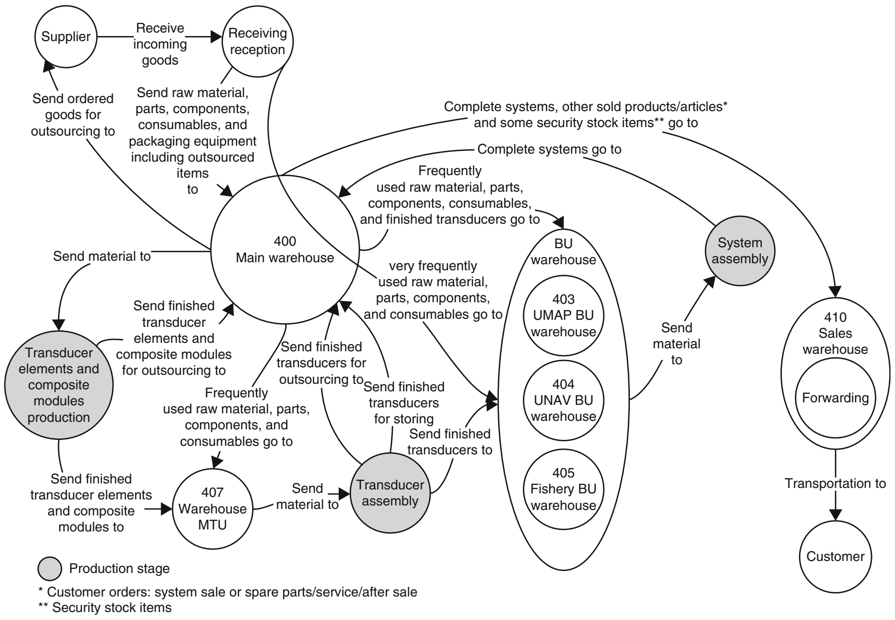
Appendix A: Excerpt from Project Swimlane Diagram

As indicated in the text, the entire swimlane diagram is extensive and company sensitive. This excerpt illustrated the use of columns, connectors, and a variety of shapes to indicate the respective activities. Likewise, portions of the total diagram were partitioned using dashed lines to reflect specific goals.


Microsoft Dynamics AX 2012
Planner MTU
Planners MSTU [BU]
Sales and Markets
Finance
Procurement
Logistics
Production
Products with very low demand and low ROP levels are automatically controlled
Available inventory
A PO picking list is generated
ROP settings
Set and maintain the ROP levels. The points are revised every 3-4 week
Group leader compare production plan, available inventory, and other data
Enough inventory to produce PO?
No actions are taken
A replenishment order is placed
The need, quantity, and timing of the replenishment order is evaluated manually in regard to the production plan
Is the replenishment order really required?
Order is cancelled
Goods are retrieved at the receiving reception
Receiving staff register the articles. Access item information
Label the articles. Print the label from AX 2012
Testing of article requested by AX 2012?
Quality check conducted
Monitor the production plans
Group leaders compare production plan, available inventory, and other data
Inventory depending on internal production?
Discuss with group leaders and take appropriate action
A replenishment order is requested
YES
NO
Procurement of articles
Registration of incoming goods
Updating and maintaining of re-order point

Appendix B: Systemigram of Company Warehouse Interactions

As stated in the text, systemigrams were an important communications vehicle for understanding the unique interactions between various warehouses in the case company.


Supplier
Receive incoming goods
Receiving reception
Send ordered goods for outsourcing to
Send raw material, parts, components, consumables, and packaging equipment including outsourced items to
Complete systems, other sold products/articles* and some security stock items** go to
Complete systems go to
Frequently used raw material, parts, components, consumables, and finished transducers go to
400 Main warehouse
Send material to
very frequently used raw material, parts, components, and consumables go to
BU warehouse
403 UMAP BU warehouse
404 UNAV BU warehouse
405 Fishery BU warehouse
System assembly
Send material to
410 Sales warehouse
Forwarding
Transportation to
Customer
Send finished transducer elements and composite modules for outsourcing to
Transducer elements and composite modules production
Send finished transducers for outsourcing to
Send finished transducers for storing
Send finished transducers to
Frequently used raw material, parts, components, and consumables go to
Send finished transducer elements and composite modules to
407 Warehouse MTU
Send material to
Transducer assembly
Production stage
* Customer orders: system sale or spare parts/service/after sale
** Security stock items

References

1. Buede DM, Miller WD (2016) The engineering design of systems: models and methods. Wiley, Hoboken
2. Bonnema GM, Veenvliet KT, Broenink JF (2016) Systems design and engineering: facilitating multidisciplinary development projects. CRC Press, Boca Raton
3. Meadows DH (2008) Thinking in systems: a primer. Chelsea Green Publishing Company, White River Junction
4. http://watersfoundation.org/systems-thinking/habits-of-a-systems-thinker/
5. INCOSE (2015) In: Walden DD, Roedler GJ, Forsberg KJ, Hamelin RD, Shortell TM (eds) Systems engineering handbook: a guide for system life cycle process and activities, 4th edn. International Council on Systems Engineering, San Diego
6. Bahill T, Brown P, Buede D, Martin JN (2002) Systems engineering fundamentals. Insight 5:7–10

7. Haskins C (2008) Systems engineering analyzed, synthesized, and applied to sustainable industrial park development. Norges Teknisk-Naturvitenskapelige Universitet, Trondheim, p 175
8. Fossnes T, Haskins C (2008) 2.4.1 an investigation into how systems engineering can help preserve natural environments. INCOSE International Symposium 18(1):280–292
9. Shainee M et al (2012) Designing offshore fish cages using systems engineering principles. Syst Eng 15(4):396–406
10. Eide HM, Haskins C (2016) Trade study of alternative controls and power distribution architecture in subsea processing. INCOSE International Symposium 26(1):2148–2163
11. Edson R (2008) Systems thinking. Applied. A primer. Applied Systems Thinking Institute, Arlington
12. Ramsay DA, Boardman JT, Cole AJ (1996) Reinforcing learning, using soft systemic frameworks. Int J Proj Manag 14:31–36
13. Boardman J, Sauser B (2008) Systems thinking, coping with 21st century problems. Taylor & Francis Group, Boca Raton
14. Sauser B, Li Q, Ramirez-Marquez J (2011) Systemigram modeling of the small vessel security strategy for developing enterprise resilience. Mar Technol Soc J 45:88–102
15. Sherman D, Cole A, Boardman J (1996) Assisting cultural reform in a projects-based company using Systemigrams. Int J Proj Manag 14:23–30
16. Blair CD, Boardman JT, Sauser BJ (2007) Communicating strategic intent with systemigrams: application to the network-enabled challenge. Syst Eng 10:309–322
17. Friedenthal S, Moore A, Steiner R (2014) A practical guide to SysML: the systems modeling language. Morgan Kaufmann
18. Rittgen P (2006) Enterprise modeling and computing with UML. IGI Global
19. Wijnhoven F (2013) Enabling the collective brain for organizations: a quickstart in management software skills. University of Twente, Faculty of Management and Governance, Department of Industrial Engineering and Business Information Systems, Enschede
20. Yin RK (2009) Case study research: design and methods. Sage Publications, Los Angeles
21. Karlsson C (2009) Researching operations management. Taylor & Francis, New York
22. Silverman D (2006) Interpreting qualitative data: methods for analyzing talk, text and interaction. Sage Publications
23. Meadows D (1997) Places to intervene in a system. Whole Earth 91:78–84
24. Eisenhardt KM (1989) Building theories from case study research. Acad Manag Rev 14:532–550
25. Kvale S (2007) Doing interviews. SAGE Publications, London
26. Edwards R, Holland J (2013) What is qualitative interviewing? Bloomsbury Academic, London
27. Ruud KS (2016) Multi-stage replenishment system: a case study. Norwegian University of Science and Technology, Department of Production and Quality Engineering, Trondheim

Chapter 76
Development of a Project-Oriented and Transnational Master Course for Training the Engineering Competencies

Cecilia Haskins, Tim Stock, Bartłomiej Gładysz, and Marcello Urgo

Abstract Mobility, multilocality, and transnational migration are current social developments among the population of the European Union. European society is becoming increasingly characterized by intercultural and cross-border interactions between citizens. This development is observable already within the activities of European companies. Cross-border project work between productions sites as well as transnational cooperation is essential for ensuring the competitiveness of the continent. These social developments in society and companies lead to new requirements for working in the European Union. Teaching and learning in higher education needs to adapt to these developments. Young engineers graduating from universities must be capable of working in international teams. In their future career, they will have to be able to work with colleagues, suppliers, and customers from different cultural backgrounds and in different countries, master the challenges of virtual cooperation in specific engineering tasks and within international value chains. As a result, new and innovative teaching and learning concepts in higher education must provide the competencies for transnational teamwork in the curriculum of tomorrow's engineers in order to ensure a competitive advantage in their future careers.

Keywords Multicultural teamwork • Multidisciplinary teamwork • Transnational cooperation • Blended learning in higher education

C. Haskins (✉)
NTNU – Norwegian University of Science and Technology, Trondheim, Norway
e-mail: cecilia.haskins@ntnu.no

T. Stock
Technische Universität Berlin, Berlin, Germany
e-mail: stock@mf.tu-berlin.de

B. Gładysz
Warsaw University of Technology, Warsaw, Poland
e-mail: b.gladysz@wip.pw.edu.pl

M. Urgo
Politecnico di Milano, Milan, Italy
e-mail: marcello.urgo@polimi.it

A.M. Madni et al. (eds.), *Disciplinary Convergence in Systems Engineering Research*, https://doi.org/10.1007/978-3-319-62217-0_76

76.1 State of the Art and Research

Reich [1] stated that conventional teaching methods do not prepare graduates to deal with problems that require them to apply their knowledge to new domains. Graduates should be prepared to perform in a turbulent, European and worldwide, transnational, and multicultural environment, which is coined continuously by new social, economic, and environmental trends promoted by globalization. As a result, it is necessary to prepare graduates for dynamic labor market demands. Thus, the requirements for teaching and learning in production and engineering management and mechanical engineering are derived from socioeconomic and sociotechnical changes observed in Europe and the world. With the intention of closing this gap, a new interuniversity master course focusing on action-based as well as on blended teaching and learning in transnational and interdisciplinary project teams is being developed and implemented on the university level under the auspices of Erasmus+ funding. The objective of the course is the development of a sustainable, technology-oriented entrepreneurial initiative. This master course called "European Engineering Team" (EET) has been jointly established by the European partner universities represented by each of the authors. It includes virtual cooperation between the students combined with intermediate face-to-face meetings in order to train mobility, individual, and intercultural competencies while working on the project.

EU's strategy *Europe 2020* reflects the importance of economic growth and creating new jobs, energy and climate change, and welfare and social security [2]. Vernon concluded that an effective learning program in engineering education should be (1) student centered and (2) project oriented and (3) include some elements of economics and management [3]. A recent Delphi study was conducted to explore the key competencies that are most relevant for sustainable development and should be undertaken for implementing higher education for sustainable development [4]. The 70 experts concluded that the most important key competencies are those for systemic thinking and handling of complexity, anticipatory thinking, and critical thinking. This is consistent with other studies that reinforce the trend to meet the complexity of today's business landscape with a team approach and to begin early to train future leaders to build effective teams [5].

The Manufacturing Institute and Deloitte Consulting LLP publish on a regular basis a Skills Gap Report that assesses the difficulties for manufacturing companies to fill critical positions [6]. When these companies were asked what they considered to be the most serious skill deficiencies in their current employees, inadequate problem-solving skills was the most frequent and relevant deficit. Skills such as critical thinking and problem solving are key competitive factors to model, analyze, and communicate information and serve as critical platforms on which leadership and entrepreneurial skills can be developed [7]. The report also identifies a set of actions to be taken in order to reduce these gaps when considering the needs of manufacturing industries, e.g., more internships and mentorships to align higher education with industry competency and skill requirements and more competency-based postsecondary pathways [8].

There are a number of initiatives related to new forms of collaboration with industry [9], including the education of future engineers. However, there are not so many multi-university and education-oriented initiatives. Ziemian and Sharma [10] addressed possibilities of utilization of so-called learning factories to develop the competencies of engineers in Europe and give the necessary priority to the transfer of technology from science to production, but they did not address the initial phases of inventing innovations, i.e., the conceptual, research, and analytical tasks necessary to be performed. Graduates should be able to act proactively to face continuous change in knowledge and technology. Therefore, the challenges of teaching engineers need to address a twofold problem typified by the questions of "what to teach?" and "how to teach?" Additionally, Jack et al. [11] identify some key actions to be accomplished in order to improve the quality and effectiveness of engineering education in manufacturing, e.g., encourage students to pursue global travels and projects, incorporate topics and courses that support global manufacturing and, in particular, encourage teaching methods that engage students.

Experiential learning results from works of John Dewey [12], Kurt Lewin [13], Jean Piaget [14], David Kolb [15, 16], and others. Individuals, especially adults, learn very effectively by reflection on what they do, in contrast to merely following instructions and procedures. Project-based learning (PBL) [17] and experimental learning laboratories are, among others, possible forms of implementing the concept of experiential learning [18]. The concept of experiential learning is illustrated in Figs. 76.1, 76.2, and 76.3.

An example of similar multi-university and education-oriented initiatives in the scope of the European Engineering Team is "Global Engineering Teams" managed by Global Education Team UG is an international and interdisciplinary project-oriented study course specifically for engineers [19]. Students from universities in different countries such as the USA, South Africa, and Brazil form one Global Engineering Team. The international and interdisciplinary group of students work throughout the course on a project provided by an industrial partner. In the UK, the University of Surrey has been teaching sustainable development to engineering

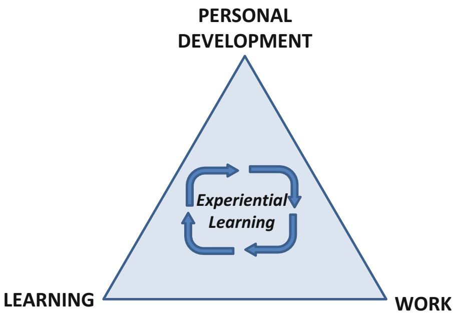


Fig. 76.1 Three aspects of experiential learning [15]

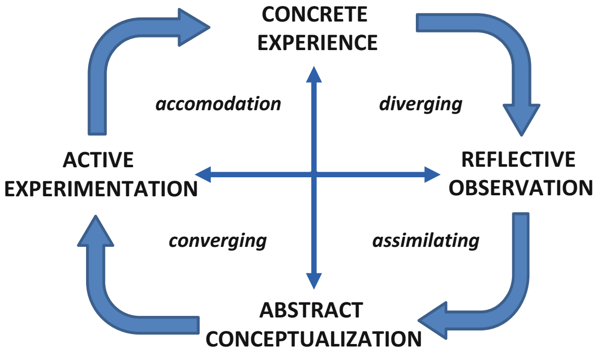


Fig. 76.2 Model of an experiential learning cycle [16]

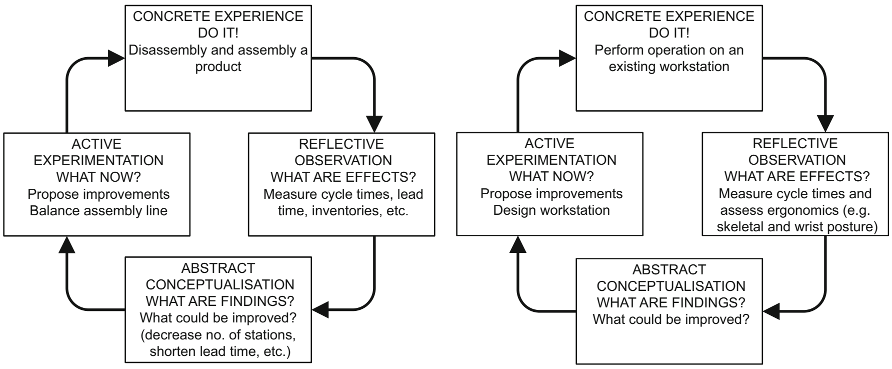

CONCRETE EXPERIENCE DO IT! Disassembly and assembly a product
REFLECTIVE OBSERVATION WHAT ARE EFFECTS? Measure cycle times, lead time, inventories, etc.
ABSTRACT CONCEPTUALISATION WHAT ARE FINDINGS? What could be improved? (decrease no. of stations, shorten lead time, etc.)
ACTIVE EXPERIMENTATION WHAT NOW? Propose improvements Balance assembly line
CONCRETE EXPERIENCE DO IT! Perform operation on an existing workstation
REFLECTIVE OBSERVATION WHAT ARE EFFECTS? Measure cycle times and assess ergonomics (e.g. skeletal and wrist posture)
ABSTRACT CONCEPTUALISATION WHAT ARE FINDINGS? What could be improved?
ACTIVE EXPERIMENTATION WHAT NOW? Propose improvements Design workstation

Fig. 76.3 Applications of Kolb's cycle [18]

students since 1998 [20]; in the USA, comparable programs have been described in Arizona [21] and Colorado [22], and most recently in Japan [23].

The EET builds upon these precursors to create a unique transnational course based on experimental learning and teaching, which expands sustainable engineering to include competitive technological innovations for empowering a global sustainable development.

76.2 Concept and Integration of the Master Course

The theme of CSER this year, disciplinary convergence, summarizes the primary goals of this project. Creation of this master course aims at increasing the higher education teaching and learning productivity of engineers by establishing an interdisciplinary and intercultural team of students and supervisors from different European universities. Each university brings in three to four master students, one professor as a supervisor, and one PhD, postdoc, or assistant professor as assistant supervisor to the course. The students of this European Engineering Team work together to devise a sustainable technology-based innovation. This innovation should demonstrate its commercial viability through the development of an entrepreneurial start-up within the last phase of the project work. The project work itself is supported by eLearning lectures addressing different topics on sustainable engineering, innovation, and virtual cooperation. Students have a certain freedom to schedule the virtual learning phases that provide the required methodological and professional competencies relevant to sustainable engineering. The course also includes four physical presence phases (i.e., face-to-face meetings) of 5 days hosted at each partner university for increasing the individual and mobility competencies of the students. In this sense, the concept of the course follows the idea of action learning in combination with blended learning. The training and teaching activities related to the master course combine theoretical knowledge, practical application and international as well as intercultural experience. Additionally, they aim to anchor sustainable thinking more deeply into the working method of the students since sustainability topics and aspects are the focus of the theoretical knowledge transfer as well as of the project work.

The integration of the master course on the university level contains five essential phases:

1. The first phase aims to incorporate the master course "European Engineering Team" into the local curriculum of each partner university. This includes the design of a university-specific module description for the course, the definition of an individual credit point structure, as well as the determination of a grading system.
2. The second phase addresses the recruitment and selection of master students at each university. Candidates include the best students who have already been working as student researchers for the participating university chairs or who have already performed outstandingly during their earlier work. Should the response exceed the capacity for available places, the students are chosen according to their competencies. Additionally, the target is to achieve an equitable gender distribution.
3. The third phase comprises the actual project work of the master students. The students consider broad thematic topics in the area of Sustainable Engineering, such as waste reduction or sustainable factories of the future, during the project kick-off-meeting. Based on these thematic topics, the students develop their own problem and solution domain by using systems engineering and design thinking methodologies. Their choice must address a technological innovation for coping with a sustainability challenge. Once a problem/solution is agreed, the students then define specific subtasks and related durations and organize themselves to achieve these objectives. One supervisor or assistant supervisor follows each subgroup. The students volunteer to participate according to their individual preferences and competencies. Ideally, each subgroup includes a student from each university to form an interdisciplinary team within the EET. A student can thereby contribute to more than one subgroup. The assignment procedure also

Table 76.1 EET eLearning topics and responsible university

Topics		Responsible university
1	Sustainable value creation	TUB
2	Systems thinking/systems engineering – a method and discipline for problem solving	NTNU
3	Technology management	WUT
4	Circular economy	POLIMI
5	Development of sustainable start-ups – integrated development of products and business models	TUB
6	Sustainable supply chain management	NTNTU
7	Virtual and augmented reality	WUT
8	Digital continuity in manufacturing systems	POLIMI

ensures that the individual workload of all students of the EET will be similar. The students of each subgroup then determine their group coordinator from within their own ranks. The coordinator is the main contact person for the supervisors/ assistant supervisors and leads the work of the interdisciplinary group. The students cooperate through virtual meetings as well as during the face-to-face working sessions scheduled over a calendar year.

4. The fourth phase runs in parallel to the previous three phases and covers the design and implementation of the eLearning content. This content provides the relevant competencies for the students' project work and covers eight thematic topics. The partners develop two lectures of 60 minutes each, with exercises. The students are able to access the lectures via a web-based platform. Table 76.1 lists the topics initially defined for this phase.
5. The fifth phase consists of performing a course assessment as well as at deriving improvement measurers for future repetitions of the master course. The supervisors and assistant supervisors create individual reviews of the course by taking into account quantitative indicators, e.g., quantity of students' applications or dropout rate, and qualitative indicators, e.g., cooperation and communication between the stakeholders or efficiency and effectiveness of the course. Moreover, each participating student can anonymously give "bottom-up" feedback using the questionnaire provided via the web-based platform about her/his experiences during the project work and s/he is invited to propose improvement measures for the next iteration of the course. After completing the project work, all supervisors and assistant supervisors carry out an assessment and performance workshop. This workshop covers the formulation of an improvement concept and concrete improvement measures agreed on by the supervisors for implementation in the next EET.

76.3 First Results of the Implementation Process

One of the auxiliary objectives of the course is to develop engineers capable of higher-level decision-making capabilities. As practicing engineers, they will need to deal with problems that are not well-defined, may have multiple solutions and are complex in nature [2]. The student activities began with understanding the problem, evaluating alternatives, then building, testing, and documenting the solution for eventual sale in a to-be-defined market. During the introductory section of the first physical transnational EET meeting, students received a brief overview in systems engineering and start-up development. These methods each provide frameworks within which the students could structure their collaboration. Systems thinking is generally acknowledged as essential for tackling the type of wicked problems normally associated with sustainability by helping to cope with the 'uncertainty, complexity, and value conflicts' associated with such problems [21, 25]. Others make a strong case for including entrepreneurial considerations as early as possible in design decisions [26].

Students shared a common attribute in that they all were recruited from programs in mechanical engineering. However, the similarities stop there as each university offers different curriculum and options for specialization. For example, within the group there were varying levels of appreciation for the concept of sustainable development and the importance of considering each of the three pillars equally: environment, economy and society. The students also varied in the amount of prior project teamwork they experienced, and their familiarity with the theory and practices of working in teams. Tuckman [27] describes the developmental stages of small groups, which he summarized as illustrated in Fig. 76.4.

Warsaw hosted the first transnational EET physical meeting in an ideal setting for brainstorming and team formation activities with a room that contained whiteboards, flipcharts, adequate seating around a large table and floor seating made comfortable by extra-large pillows. In the course of the week, the students availed themselves of all these features. The room also allowed the student advisors to sit out of the way but in a position to observe the group dynamics and progress. The observations throughout the week were captured and are summarized in Appendix A.

Later review of the overall progress confirmed that the Tuckman framework (Fig. 76.4) provides a good indication of the evolving maturity of the group and forecast a positive outcome as they formed smaller teams for their information gathering before the second face-to-face meeting. Students followed techniques for nominal consensus building to ensure that they could identify an appropriate set of activities by the end of the first week [30]. These practices will be continued as they

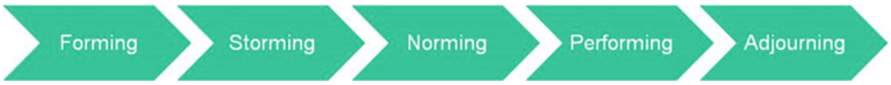


Fig. 76.4 Tuckman's five stages of teamwork maturity

move to agree upon which solution(s) are worthy of prototyping and begin to use the innovation diamond to identify potential markets for the solution. During the third face-to-face, the students built working prototypes of their products and attended a business modeling workshop. The workshop was customized to address their products and the challenges of bringing them to market. By their final gathering, the students were well acquainted with each other and worked efficiently to create their final presentations and course report.

76.4 Systems Engineering: Systems Thinking

The most honest appraisal of this first cohort would be to describe it as a seemingly random walk. Much of this observation is related to the freedom given to the students as follows:

- Free to organize work, groups – restructure their groups – self-empowerment, management
- Free to make wrong decisions
- Free to select their own topic within sustainability

In addition, the students were challenged by the following situations:

- Students had uneven engineering skill sets, even from within same university
- Students were unfamiliar and inexperienced with distance coordination
- Students were mostly unfamiliar with sustainability concepts – a built in expectation
- Students were in their first position of taking decisions that carried international relevance
- Before the third workshop in Berlin in October, none of the students understood what it meant to adopt an entrepreneurial mindset – another built in expectation of the project
- Students became distracted by the prototype – something physical that they could measure
- None of the students were working in their first language – the working language was English
- Students were expected to travel to each campus and they experienced mobility challenges
- The universities had different academic calendars which caused scheduling conflicts – one set of students free to work on the project while another set were taking exams or on vacation
- The students were expected to use the EET specific platform, which was unfamiliar to everyone

Given this high level of uncertainty, the students managed fairly significant personal and team growth. The project conditions simulated real-world on-the-job conditions, and they learned to use participatory democracy techniques to make decisions. A brief introduction to the rudiments of systems engineering for problem solving helped them in their first meetings, but their fallback process reduced to trial and error for much of the engineering decisions.

In light of these observations, the supervisors will adjust their direction and support to the second cohort – beginning in April 2017 – as follows:

1. Students need more structure from supervisors, while retaining their freedom to learn and make mistakes.
2. Students need to work in smaller consistent teams for the entire duration of the project.
3. Students needed to be less specialized – everyone should be concerned about business and sustainability matters, not just engineering.
4. Academic calendars can create transnational disconnects, so these have been carefully coordinated.
5. Students need to be motivated to use the EET platform to increase transparency.
6. Students should be advised that there is a learning curve for finding a good way to "skype" and manage distance communications.
7. Supervisors will retain the total group interim meetings to facilitate cooperation between the physical meetings.
8. Faculty from each university achieved some insights into curriculum programs across Europe.

In summary, the first cohort was praised for their fortitude and perseverance to complete the project, and for their fine results. In their final report and presentation, they confirmed many of the observations noted above. As a result, the project should appear more polished for the second cohort. The combined experiences will be used to create guidelines for future implementations. The university partners are still committed to this format and already discussing ways to move forward.

76.5 Summary

In conclusion, the results of this project will provide a novel concept for a project-oriented and interuniversity engineering master course supported by eLearning. Subsequently, the first results of the implementation will serve as an initial template for the EET course, which will be further tested in a second cohort. In addition, a full validation of the 2 $\times$ 12 months EET will be conducted for providing a final implementation guideline for the master course for the partners and other interested universities. This master course aims to prepare students to thrive in and contribute to an increasingly demanding work life in Europe by promoting key skills required in the EU labor market.

Acknowledgments This research was supported by the ERASMUS+ Strategic Partnerships Project "New Culture in Higher Education: Project-Oriented Learning Beyond Borders" funded by the European Union, project number 2015-1-DE01-KA203-002207.

Appendix A: Observations of Student Activities and Maturity Phases During Their First Meetings

Tuckman	Day	Nr.	Observation
Forming	1	1	Opening presentations from coordinator and supervisors
Storming		2	After lunch, discussion of the alternative "solutions" which did not satisfy the sustainability criterion; changed the conversation to "what is the problem?" which resulted in a decision to settle on waste management after establishing a context based on the UNSDG [28]
	2	1	Presentation on the innovation diamond
Norming		2	Agreed to continue analysis of waste management using the EU waste hierarchy [29]); create a matrix of solution ideas mapped to the hierarchy
		3	Do research to better understand the matrix
		4	After lunch, reviewed the matrix and voted for "preferred" problem domain
		5	Performed a competency inventory of the group members; agreed on criteria for evaluation of alternative solutions; weighted the criteria
		6	Lorenzo proposed that some of the solution candidates could be merged
		7	Vote for preferred solution domain – agreed on the top 3
		8	Created research groups based on the top 3 – i.e., smart cities, expiry dates on food, packaging – and look for synergies between the groups
	3	1	Groups shuffled to bring new eyes to the topic areas
		2	Groups were encouraged not to converge too early; to use the 3 areas to maintain a multidimensional perspective on the product/problem
		3	Student presentations by each of the 3 groups; Lorenzo, Marta, Kjersti, Joanne (municipal solid waste); Teo, Paulina, Kata, + (handling food waste); Even, Henry, Matt, Jo (packaging)
		4	Feedback from supervisors – look for business drivers and other driving factors that create waste; solve a problem, not a symptom
Norming	4	1	Effectively worked without any supervisor presence
Performing	5	1	Students had identified one student to serve as a project manager; they then used a democratic process to volunteer/assign group participants
		2	These newly defined groups then worked in the morning to prepare their task lists and make detail plans leading to next face-to-face in Milan
		3	Before breaking for lunch, created an overall plan leading to end in January
		4	Started right after lunch with group presentations; <u>Henry</u> started with a quick review of their process thus far
			Group 1 – <u>Marta</u>, Felix, Giovanna – Food waste; supervisor Marcello
			Group 2 – <u>Joanna</u>, Even, Lorenzo – Package waste; supervisor Bart

(continued)

Tuckman	Day	Nr.	Observation
			Group 3 – Teo, Alice, Maciek – City waste; supervisor Tim
			Group 4 – Paulina, Katarzyna, Kjersti – Home waste; supervisor Cecilia
		5	Dialogue between supervisors and the whole group
		6	Individual meeting of group members and their supervisors
		7	Adjourn

References

1. Reich R (1990) Redefining good education: preparing students for tomorrow. In: Bacharach SB (ed) Education reform: making sense of it all. Allyn and Bacon, Boston
2. European Commission (2010) Communication from the commission – Europe 2020 – a strategy for smart, sustainable and inclusive growth, Brussels.
3. Vernon J (2000) Engineering education: ending the centre or back to the future. Eur J Eng Educ 25(3):215–225
4. Rieckmann M (2012) Future-oriented higher education: which key competencies should be fostered through university teaching and learning? Futures 44(2):127–135
5. Sjøvold E (2014) Introduction to the special issue leadership and the group. Small Group Res 45(4):367–375
6. Manufacturing Institute, Deloitte Consulting LLP (2016) Skills gap in manufacturing. www.themanufacturinginstitute.org/Research/Skills-Gap-in-Manufacturing/Skills-Gap-in-Manufacturing.aspx. Viewed 26 May 2016
7. Neubert JC, Mainert J, Kretzschmar A, Greiff S (2015) The assessment of 21st century skills in industrial and organizational psychology: complex and collaborative problem solving. Ind Organ Psychol 8(02):238–268
8. Deloitte Consulting LLP, Manufacturing Institute (2015) Boiling point? The skills gap in U.S. manufacturing. www.themanufacturinginstitute.org/~/media/827DBC76533942679A15EF7067A704CD/2015_Skills_Gap_Report.pdf. Viewed 26 May 2016
9. Paci AM, Lalle C, Chiacchio MS (2013) Education for innovation: trends, collaborations and views. J Intell Manuf 24(3):487–493
10. Ziemian CW, Sharma MM (2007) Adapting learning factory concepts towards integrated manufacturing education. Int J Eng Educ 24(1):199–210
11. Jack H, Mott R, Raju V, Conkol G, Stratton M, Waldrop P, Wosczyna-Birch K, Bates S (2011) Curricula 2015; a four year strategic plan manufacturing education, SME Education And Research Community, lulu.com
12. Dewey J (2009) Education and experience. Simon and Schuster, New York
13. Schein EH (1996) Kurt Lewin's change theory in the field and in the classroom: notes toward a model of managed learning. Syst Pract 1996(1):27–47
14. Piaget J (1964) Development and learning. In: Ripple RE, Rockcastle VN (eds) Piaget rediscovered: a report on the conference of cognitive studies and curriculum development. Cornell University, Ithaca, pp 7–20
15. Kolb DA (1984) Experiential learning: experience as the source of learning and development. Prentice Hall, Englewood Cliffs
16. Kolb AY, Kolb DA (2005) Learning styles and learning spaces: enhancing experiential learning in higher education. Acad Manag Learn Edu 4(2):193–212
17. Chiriac EH (2008) A scheme for understanding group processes in problem-based learning. High Educ 55(5):505–518

18. Gladysz B, Santarek K (2014) Experiential learning for production and engineering management studies – a case study. Conf. Proc. IV international scientific-technical conference MANUFACTURING 2014, Poznan, pp 87–95
19. Global Education Team UG (2016) Global engineering teams. www.global-engineering-teams.org/. Viewed 26 May 2016
20. Perdan S, Azapagic A, Clift R (2000) Teaching sustainable development to engineering students. Int J Sustain High Educ 1(3):267–279
21. Brundiers K, Wiek A, Redman CL (2010) Real-world learning opportunities in sustainability: from classroom into the real world. Int J Sustain High Educ 11(4):308–324
22. Blizzard J, Klotz L, Pradhan A, Dukes M (2012) Introducing whole-systems design to first-year engineering students with case studies. Int J Sustain High Educ 13(2):177–196
23. Haruyama S, Kim SK, Beiter KA, Dijkema GP, De Weck OL (2013) A new project-based curriculum of design thinking with systems engineering techniques. International Journal of System of Systems Engineering 4(2):162–173
24. Claesson AN, Svanström M (2015) Developing systems thinking for sustainable development in engineering education. In: Proceedings of EESD'15
25. Lönngren J, Svanström M (2015) Systems thinking for dealing with wicked sustainability problems: beyond functionalist approaches. In: Proceedings of EESD'15
26. Young A (2015) Engineering sustainability markets: why T-shaped engineers need communication design. In: Proceedings of EESD'15
27. Tuckman BW (1965) Developmental sequence in small groups. Psychol Bull 63(6):384
28. UNSDG (2015) The UN Sustainable Development Goals (SDG). www.un.org/sustainabledevelopment/sustainable-development-goals/. Viewed 26 May 2016
29. Gharfalkar M, Court R, Campbell C, Ali Z, Hillier G (2015) Analysis of waste hierarchy in the European waste directive 2008/98/EC. Waste Manag 39:305–313
30. Hutchings HA, Rapport FL, Wright S, Doel MA (2013) Obtaining consensus from mixed groups: an adapted nominal group technique. British Journal of Medicine and Medical Research 3(3):491

Chapter 77
The Role of Decision Analysis in Industrial and Systems Engineering Education

Ali E. Abbas and Maximilian Zellner

Abstract This paper provides a pilot study on the discrepancy between current Industrial and Systems Engineering (ISE) university programs and industry requirements. A summary of an interview series conducted with subject matter experts in ISE emphasizes the need for more instruction of decision analysis in ISE curricula. According to the analysis of the ten top-ranked ISE graduate and undergraduate programs in the USA, less than 20% of the ISE programs and specializations have core classes in decision making and decision analysis. The results of the interview series and the analysis of university ISE programs illustrate the need for more decision analysis education both in undergraduate and graduate ISE education.

77.1 Introduction

In an increasingly complex world, the ability of Industrial and Systems Engineers to make qualified and scientific decisions is becoming a central hiring criterion for companies. If one is searching on the internet for a definition of Industrial and Systems Engineering (ISE), a vast set of definitions can be found. According to the Bureau of Labor Statistics, ISE deals with finding ways to eliminate wastefulness in production processes. Industrial and Systems Engineers devise efficient systems that integrate workers, machines, materials, information, and energy to make a product or provide a service [1].

Decision analysis [2] has a strong connection with Industrial and Systems Engineering. Former Secretary of the Treasury and former President of Harvard University Lawrence Summer observes that

> *…in an early era when people were involved in surveying land, it made sense to require that almost every student entering a top college know something of trigonometry. Today, a basic grounding in probability, statistics and decision analysis makes far more sense.* [3]

A.E. Abbas (✉) • M. Zellner
Epstein Department of Industrial and Systems Engineering, University of Southern California, Los Angeles, CA, USA
e-mail: aliabbas@usc.edu; mzellner@usc.edu

© Springer International Publishing AG 2018
A.M. Madni et al. (eds.), *Disciplinary Convergence in Systems Engineering Research*, https://doi.org/10.1007/978-3-319-62217-0_77

In a study conducted in 2013, Bethlehem, PA., based National Association of Colleges and Employers (NACE) surveyed 200 different employers about the skills they are looking for in recent college graduates.

The study found that the most important skill in a new hire is the ability to work in a team, and immediately in the second place are decision making and analysis capabilities [4].

Yannis Yortsos, Dean of the Viterbi School of Engineering at the University of Southern California, defines technology as the exploitation of phenomenon for useful purposes. Building on this definition, Abbas [5] defines Industrial and Systems Engineering as "… the set of decision making tools needed to move from phenomena to value."

Using the definition by [5], the value one strives to achieve can take on many forms, such as company profit, environmental protection, or even personal well-being (see Fig. 77.1).

In order to create value, we need to define what it constitutes, as it by itself might be different depending on the decision maker. A profit-maximizing firm defines value as profit and thus makes decisions that contribute to profit maximization. The assessment of the expected utility of profit, in turn, requires demand estimation [6]. Decision making to produce value also appears at the more operational level, such as when designing a value function to determine the parameters of high-speed machining facility [7].

Transforming the phenomenon in the "operating environment" in a way that achieves the desired output and value requires a multitude of decisions in an environment that involves multiple stakeholders and multiple teams. In such large systems and enterprises, we thus need to understand the ramifications of setting

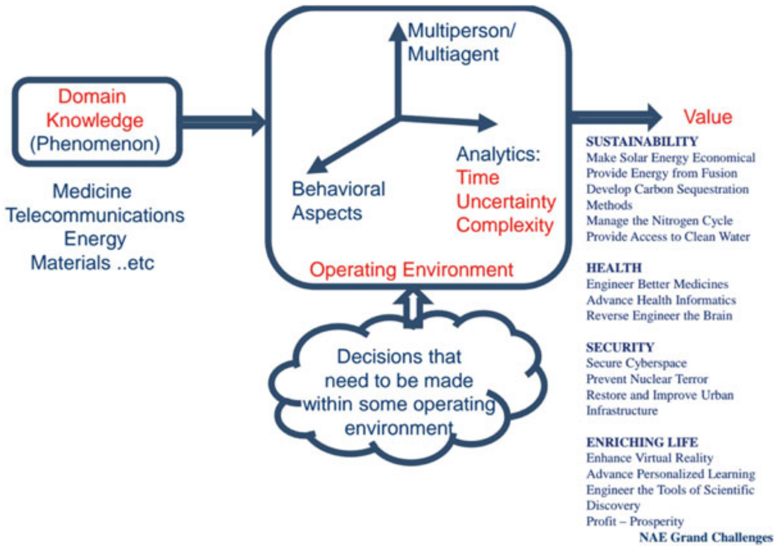

Domain Knowledge (Phenomenon)
Medicine
Telecommunications
Energy
Materials ..etc
Multiperson/ Multiagent
Behavioral Aspects
Analytics:
Time
Uncertainty
Complexity
Operating Environment
Decisions that need to be made within some operating environment
Value
SUSTAINABILITY
Make Solar Energy Economical
Provide Energy from Fusion
Develop Carbon Sequestration Methods
Manage the Nitrogen Cycle
Provide Access to Clean Water
HEALTH
Engineer Better Medicines
Advance Health Informatics
Reverse Engineer the Brain
SECURITY
Secure Cyberspace
Prevent Nuclear Terror
Restore and Improve Urban Infrastructure
ENRICHING LIFE
Enhance Virtual Reality
Advance Personalized Learning
Engineer the Tools of Scientific Discovery
Profit – Prosperity
NAE Grand Challenges

Fig. 77.1 From phenomena to value according to [5]

incentives and their effects on decision making. Incentive schemes that are not aligned with the overall corporate value function result in suboptimal and even very negative results for the company [8].

Especially in engineering and systems design, which is one of the main areas of Industrial and Systems Engineering, the important role of decision making has been regularly emphasized [9–15].

This paper provides an initial investigation into the role of decision analysis in the current Industrial and Systems Engineering curricula using two methods: (i) interviews with subject matter experts in ISE with a background both in academia and business and (ii) analysis of decision making and decision analysis education in top ISE programs.

The subject matter experts, who were asked to assess the current university ISE programs, were participants of the NSF workshop "Perspectives on the Use of Decision Analysis in Systems Engineering," held in Arlington, Virginia, in October 2015. Participants included university professors as well as professionals at several high-tech companies. The answers were obtained either by conducting 20-minute face-to-face interviews or by an exchange of emails, using a questionnaire with the exact same seven questions in Sect. 77.2. About 20% of the responses were from industry professionals, and so the responses were skewed in favor of the academic perspective, although several professors accumulated substantial industry insights through consulting projects and/or previous roles in engineering companies.

To assess the top ten university ISE programs from which to conduct our search, the annual ranking according to [16] has been used. Although each of the ranked universities offers an ISE education, the program names and curricula differ. Some program might be named "Industrial and Systems Engineering," whereas another one offers a choice between specializations in healthcare, statistics and quality, and financial engineering, for example. These specializations, including the term "General" for programs without a preliminary focus, were used to group in Sect. 77.3. In a subsequent step, the official handbook of each ISE program and specialization has been analyzed using the search terms "Decision analysis," "Decision making," and "Decision Theory" with regard to the specialized field. Depending on the specialization of the program, the search terms included, but were not limited to, combinations of the abovementioned terms with subjects like "Finance," "Financial," "Design," "Engineering Management," and "Systems Engineering." From each handbook it also became evident whether a listed course in decision analysis and decision making was considered compulsory or an elective. Thus, this approach explicitly disregarded potential courses in decision analysis and decision making that were not offered at the respective engineering schools, which usually house ISE programs.

Table 77.1 Questionnaire questions

No.	Question
1	What is your definition of Industrial and Systems Engineering?
2	What skills and tasks (functional, social, knowledge-based) do you expect of an Industrial and Systems Engineer? In the first five years? In years five to ten?
3	Do you think there is a mismatch between curricula of Industrial and Systems Engineering university programs and organizational needs? If yes, please specify:
4	In which fields do you think Industrial and Systems Engineers usually end up after graduation?
5	What are the main tasks and responsibilities of the Industrial and Systems Engineers you employ?
6	What do you consider "must-have" courses in every Industrial and Systems Engineering undergraduate and graduate curricula, respectively?
7	Where do you see the biggest work challenges facing Industrial and Systems Engineers in the immediate (up to two years) and long-term future (>2 years)?

77.2 Survey of Subject Matter Experts in Industrial and Systems Engineering

The goal of the conducted interviews with experts with academic and industry experience was to identify qualification gaps and to gain better insight into the situation of Industrial and Systems Engineering today. Table 77.1 lists the seven questions on questionnaire that were asked.

1. What is your definition of Industrial and Systems Engineering?

Approximately 25% of study participants consider Industrial and Systems Engineering as a blend between engineering and business/management disciplines, synthesizing engineering and business principles to create valuable and efficient technological and socioeconomic systems. Another one-fourth of interviewees view ISE more as a way or process to structure problems, so that they can be tackled incoherently. This approach was also circumscribed as the V-model or "Decomposition – Re-composition" of a problem. The majority (with approximately 37%) described ISE as the process to design, develop, enable, and improve a system, in order to yield a product of value. The remaining 13% of respondents view the field and profession as the guidance, coordination, and facilitation of design engineering teams in the development of a large complex engineering system.

According to one study participant, Industrial and Systems Engineering is currently moving slowly toward an incorporation of decision making into an academic field, which used to focus solely on problem-solving and optimization.

2. What skills and tasks (functional, social, knowledge-based) do you expect of an Industrial and Systems Engineer (in the first five years, in the years five to ten)?

100% of participants in the study agreed that an ISE graduate does not only need to possess methodological qualifications, for example, in mathematics, probability and simulation techniques, but also so-called soft skills, such as the ability to work in cross-functional teams.

An essential skill for Industrial and Systems Engineers is the ability to think in systems terms in order to assess the desired and undesired consequences of their own behavior and actions.

In a highly complex environment, graduates need to be comfortable with transforming real-world problems into models that can subsequently be simulated in order to determine the most important problems in a system and to devise appropriate solutions. This ability requires new hires to possess a profound knowledge of mathematics, probability, as well as procedures on how to conduct sound experiments to gather data. At the same time, Industrial and Systems Engineers need to exhibit a proficiency in value-based decision making, so that overall goals are achieved, without compromising on organizational and societal values.

According to the respondents, Industrial and Systems Engineers are strongly specialized in their fields, but at the same time they need to have the technical and social skills necessary to function well in interdisciplinary teams.

In order to experience a steep learning curve, which is becoming more and more important in today's work environment, ISE graduates need to have a willingness to show their ignorance and a readiness to keep learning throughout their career. As a company it "...is very cumbersome to determine in which areas and fields the new hire needs more support and guidance, which is made even more difficult if he or she is not open or willing to voice any shortcomings."

3. Do you think there is a mismatch between curricula of Industrial and Systems Engineering university programs and organizational needs?

During the interview sessions, approximately 90% of interviewees identified several mismatches between university ISE education and what organizations actually need. Only one of the study participants did not perceive any gap between university ISE education and industry requirements.

In their eyes, the current curricula in ISE show students how to solve standardized problems, which are far from what they can expect to be working on once they enter the workforce.

At the same time, they are neither equipped to formulate real-life problems nor are the standard problem-solving procedures graduates learned during their time at university very efficient at solving them. In fact, graduates are considered badly prepared for identifying and for deciding what the underlying problem is, as most of them leave university with little aptitude in understanding the overarching systems and in scientific decision making.

At the same time, study participants were aware of the fact that the extent to which some subjects can be taught at university is limited, mainly because of subject-inherent constraints. For example, one interviewee thinks that design is not being taught sufficiently in ISE and systems engineering education due to its complexity and time intensity. Another participant added that a certain mismatch between university programs and organizational needs persists in all subject areas and is not inherently Industrial and Systems Engineering specific.

4. In which fields and departments do you think Industrial and Systems Engineers usually end up after graduation?

The expert interviews confirmed what the Bureau of Labor Statistics already stated. All of the interview participants agree that Industrial and Systems Engineers are qualified to join diverse departments after their graduation from university.

In their opinion, most graduates are very likely to work in manufacturing or manufacturing-related professions, such as quality and process performance improvement, as well as in research and development, systems architecting and integration, and supply chain management.

As these departments are necessary in almost every business, Industrial and Systems Engineers can and will be found in industries ranging from oil extraction, over manufacturing and healthcare, to software corporations and consulting. According to one participant from an industry, most ISE graduates end up in professions that are more related to the industrial part of their education. In his opinion, the aerospace industry sources most of its systems engineers from traditional engineering disciplines, such as electrical, mechanical, aerospace, and software engineering, thus foregoing the benefits that come from hiring an academically trained systems engineer.

5. What are the main tasks and responsibilities of the Industrial and Systems Engineers you employ?

Interviewees agreed that Industrial and Systems Engineers' tasks are twofold. In their daily routines, Industrial and Systems Engineers usually deal with every task that comes with business process improvement projects. Not only are they responsible for the analysis and development of the underlying process but also for system integration, necessary facility layout and design, and accompanying risk assessments.

In case Industrial and Systems Engineers are working more closely with potential customers, their "…role is also in product architecting and instrumentation…" in order to deliver maximum value, according to one respondent.

6. What do you consider "must-have" courses in every Industrial and Systems Engineering undergraduate and graduate curricula, respectively?

Approximately 80% of respondents replied that not only should universities teach their students a more practical problem-solving approach but also increase their decision-making capabilities as well as their holistic system thinking skills. At the same time, all of them agreed that students need to acquire a solid background in mathematical modeling and simulation, probability and statistics, operations research, project management, as well as design. For them to be properly prepared for the work environment, universities also need to put an emphasis on cross-functional team projects.

In order to increase students' ability to gather and analyze the ever-increasing amount of data, 20% of subject matter experts also suggest that universities should include more computer science courses in Industrial and Systems Engineering curricula.

7. Where do you see the biggest work challenges facing Industrial and Systems Engineers in the immediate and long-term future?

Subject matter experts agreed that the biggest challenges to the field are how to become and stay relevant to business, maintaining and improving knowledge about more and more complex systems and avoiding an obsession with procedures and practices that obscures actual engineering practice.

As the amount of data and complexity of systems increases, it is going to be more difficult for the profession to stay relevant and not to lose it to more data-driven fields like computer science. At the same time, Industrial and Systems Engineers need to become more articulate about the importance of their services in product design, development, and improvement, if they want to retain their relevance to business. In the long run, system complexity is going to make it more strenuous to identify interactions and uncover the underlying problems of systems, which in turn makes it almost impossible to decide which problem to solve first, if you do not follow a stringent scientific approach.

77.3 3. Insights from Interview Results

The main findings of the interview series can be summarized as follows:

- In general, the study participants do not share a common definition of Industrial and Systems Engineering. Whereas one group of participants uses an abstract definition of it being a blend between engineering and management, others see it as a problem-solving approach, a process to design, develop, and improve systems to yield a product of value, or as the guidance, coordination, and facilitation of design engineering teams.
- According to all study participants, Industrial and Systems Engineers must possess a solid foundation of mathematics, probability and statistics, modeling and simulation, as well as preferably computer science skills. At the same time, they consider soft skills, such as a readiness to learn and to work in teams, essential for being effective in the workplace. These skills are all necessary to understand a complex system in depth and to develop according solutions.
- Approximately 75% of respondents emphasized the importance of decision analysis and decision making in systems engineering. Of all the interviewees who perceived a mismatch between university ISE education and organizational needs, 70% mentioned that decision analysis and decision making need to be taught more thoroughly, in order to prepare graduates for the work environment.

One can see that these areas are interdependent. A scientific approach to decision analysis requires the gathering of information in order to assess probabilities of prospects. Also, the results of a decision are influenced by the extent to which the decision maker understands the system he or she is making the decision in. If his or her mental system representation does not match reality, the outcome might be affected negatively.

In the next part of this paper, the ISE programs at the ten best-ranked universities are analyzed for their teaching of decision analysis and decision making.

77.4 4. Survey of Top Ten ISE University Programs

Following the results from the interview series with experts in the Industrial and Systems Engineering field, we explored the ISE programs at the ten top-ranked institutions in the ISE field with regard to their teaching of decision analysis. The best schools to study ISE, according to [16], are depicted in Table 77.2.

The first step of the analysis focused on all available undergraduate and graduate programs in the field of Industrial and Systems Engineering at abovementioned universities. At the ten best-ranked universities in the country, one can find a total of 136 different programs and specializations, with

- 35 undergraduate
- 58 graduate (Master)
- 43 graduate (PhD)

level programs. Because of the wide field of application in different industries and organizations, Industrial and System Engineering programs at universities offer students a multitude of different focus areas. As a result, studying Industrial and Systems Engineering at any of the ten top-ranked universities can mean a multitude of different programs, potentially resulting in very different qualification levels of graduates. As a potential ISE student, one has the choice between programs that try to cover a wide field of topics and ones that focus on distinct areas from the

Table 77.2 Best ranked universities for Industrial/Manufacturing/Systems Engineering according to [16]

Rank	University
1	Georgia Institute of Technology
2	University of California – Berkeley
2	University of Michigan – Ann Arbor
4	Northwestern University (McCormick)
4	Stanford University
6	Massachusetts Institute of Technology
7	Cornell University
7	University of Wisconsin – Madison
9	Purdue University – West Lafayette
9	Virginia Tech

Table 77.3 University focus areas/specializations ranked by percentage of undergraduate programs in each area

Undergraduate ISE programs (Total:35)	
Operations research	17.1%
Engineering management	14.3%
Decision analysis	11.4%
Economics/finance	11.4%
General	11.4%

Table 77.4 University focus areas/specializations ranked by percentage of graduate (Master) programs in each area

Graduate (Master) ISE programs (Total: 58)	
Operations research	19%
Operations (production, logistics, etc.)	13.8%
Engineering management	12.1%

beginning. Looking at the program details of each of the best-ranked universities, one can conclude that they are giving their students the possibility to focus their interest on a specific topic already during undergraduate education. On the one hand, this early focus might be beneficial for companies whose requirements for new hires are getting ever more specialized. On the other hand, this development makes it ever more complicated to discern what Industrial and Systems Engineering programs really need to cover and whether some forms omit elements that need to be a standard component of every ISE curriculum.

As one can see from Table 77.3, the biggest single undergraduate focus in Industrial and Systems Engineering is operations research with 17.1% of all 35 undergraduate programs analyzed, followed by engineering management (14.3%), and the areas such as decision analysis, economics/finance, and general (tied with a share of 11.4%).

In graduate (Master) programs, decision analysis does not make it into the biggest three, which are operations research (19%), operations (13.8%), and engineering management (12.1%) (see Table 77.4).

Considering the graduate (PhD) programs though, decision analysis shares the rank of most common Industrial and Systems Engineering focus area with engineering management and operations research (see Table 77.5).

Given this insight from Tables 77.3, 77.4, and 77.5, one might assume that universities offer a fair share of programs with a focus on decision analysis, as this focus area is one of the biggest in undergraduate and graduate (PhD) programs (in graduate (Master) programs this area ranks 7th out of eight). But having determined decision making as an essential ability hiring companies are looking for in graduates, this research also searched the curricula of each of the 136 programs and determined whether all students, no matter what their primary focus area is, can take decision analysis electives or whether they are required to take at least one decision analysis or decision-making class.

Table 77.5 University focus areas/specializations ranked by percentage of graduate (PhD) programs in each area

Graduate (PhD) ISE programs (Total: 43)	
Decision analysis	11.6%
Engineering management	11.6%
Operations research	11.6%
Human factors	9.3%
Quality and statistics	9.3%
Computer science	7%
Economics/finance	7%
Healthcare	7%
Operations	7%
Policy	7%

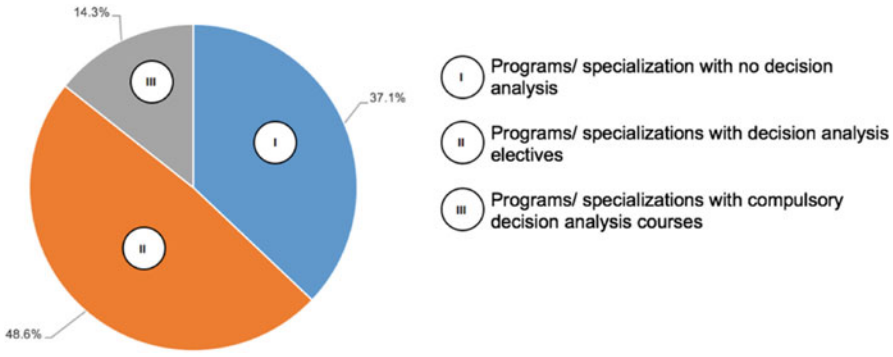


Fig.77.2 Share of decision analysis in undergraduate ISE education

On an undergraduate level, 37.1% of the ISE programs offered do not have any electives or compulsory courses in the field of decision analysis. Despite a comparatively high share of 48.6% of undergraduate programs that offer such electives, 85.7% of undergraduates may not receive adequate training in decision making (see Fig. 77.2).

The same is true for Master level graduate courses, which can be seen in Fig. 77.3. Roughly 10% of all programs have compulsory decision analysis courses, whereas 89.7% offer only electives or no such classes at all.

The increase in programs that do not offer any decision analysis courses at all at PhD levels compared to undergraduate and graduate (Master) levels might be explained with the less generalist character of PhD programs. Only 18.6% of PhD programs offer decision analysis and decision making courses, with 67.4% with no courses at all and 14.0% with electives, as depicted in Fig. 77.4.

A significant percentage of university programs do not make it compulsory for students to acquire decision analysis and decision making skills, which is a stark contrast to the skill set that experts recommend graduate students have, if they want to succeed in the workplace.

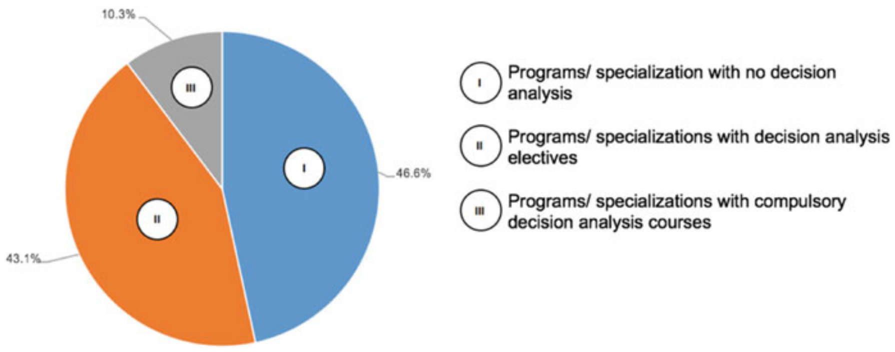


Fig.77.3 Share of decision analysis in graduate (Master) ISE education

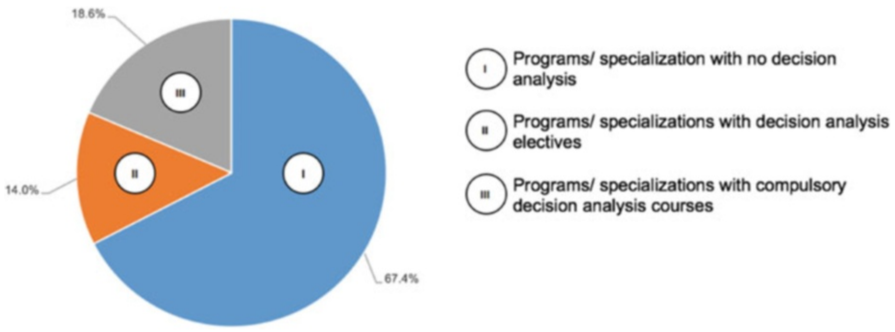


Fig.77.4 Share of decision analysis in graduate (PhD) ISE education

Thus, given this data and the expert opinion in part II, one can see that the current ISE education at the top ten US universities does not prepare their graduates sufficiently for the work environment when it comes to decision analysis.

The following and concluding part is now going to summarize the finding of this paper and offer recommendations on how universities can adapt their ISE curriculum accordingly, so that graduates are best prepared for joining the Industrial and Systems Engineering workforce.

77.5 Conclusion and Recommendations

According to several surveys of the needs of companies that are looking for new employees, decision making and decision analysis is regarded as one of the most important skills. According to subject matter experts it is even going to become more important, as systems get more complex, where poor decisions are more likely to result in unpredictable consequences. Thus it can be perceived as a responsibility of a society's educational system to prepare ISE graduates for these challenges with a solid background in decision analysis and decision making.

The conducted interviews for this paper were part of a pilot study and need to be extended to more participants in a follow-up investigation. Nonetheless, the conducted interview series with experts with backgrounds both in industry and academia has uncovered that the current approach of educating Industrial and Systems Engineers does not suffice to prepare graduates for the quality decision making of the modern-day work environment. According to the experts' view, new graduates are not ready to solve real-world problems, as they are only faced with artificial problems that are helpful for demonstrating how a specific optimization or problem-solving approach works.

Furthermore, the experts identified a shortcoming in graduates' ability to identify the most relevant problem, as universities do not include sufficient decision making or decision analysis classes in their curricula. This finding is also backed by the analysis of the 136 Industrial and Systems Engineering programs and specializations at the ten best-ranked universities in the country, which showed that only 14.3%, 10.3%, and 18.6% of programs had compulsory decision analysis courses on undergraduate, graduate (Master), and graduate (PhD) levels, respectively.

For graduates to have a solid ISE knowledge, they need to understand the interaction of the more strategic part of decision making and the more operational focus of optimization. As a matter of fact, there seems to be a confusion of operations research with optimization as well. According to [17], decision analysis is a fundamental part of operations research, but very little of the operations research programs at the top ten ISE colleges actually offer courses in it. Instead they focus more on optimization issues.

To prepare students better for what is going to await them in the work environment, universities should start incorporating core decision analysis and decision making courses in each undergraduate and graduate level Industrial and Systems Engineering program and specialization they are offering. As Summers already implied in his article, decision making should be a necessary pillar of our educational system nowadays [3].

As this paper has put an emphasis on the divergence between university programs and employer needs regarding decision making, there is no reliable data yet on how the programs are going to perform using other necessary metrics, such as the ability of system thinking, data sciences, and analytics exposure, and the incorporation of soft skill components into their curriculum. During the conducted interviews though, these issues have been identified as shortcomings of most of today's Industrial and Systems Engineering university programs. Thus, future work will cast a light on how universities teach graduates these skills today and whether there are opportunities of improvement. Nevertheless, Industrial and Systems Engineering programs should consider this expert advice and include courses that enhance their students' ability to work in cross-functional teams on real-life problems, to start thinking in and modeling interrelated systems, as well as helping them build their data gathering and analytics skills by incorporating selected computer science courses into their curriculum.

Acknowledgment This work was supported by the National Science Foundation awards CMMI 16-29752, CMMI 15-65168, and CMMI 16-44991. The views in this work are those of the authors and do not necessarily represent the views of the National Science Foundation or the Federal Government.

References

1. Bureau of Labor Statistics. Occupational Labor Statistics, E-source.http://www.bls.gov/ooh/architecture-and-engineering/industrial-engineers.htm#tab-4, Oct2016.
2. Howard RA, Abbas AE (2015) Foundations of decision analysis, 1st edn. Prentice Hall, Upper Saddle River
3. SummersL (2012)What you (really) need to know. New York Times, 20Jan2012
4. AdamsS(2013)The 10skills employers most want in 20-something employees.www.forbes.com/sites/susanadams/2013/10/11/the-10-skills-employers-most-want-in-20-something-employees/#7711d2c752d5
5. AbbasAE (2016) Perspectives on the use of decision analysis in systems engineering: workshop summary, IEEE systems conference (SysCon), 15Apr2016, Orlando,Florida
6. AbbasAE, HazelriggG, Al-KindiM (2012)Bayesian inference for the demand of engineering products.ASME DTM International design engineering technical conferences and computers and information in engineering conference
7. Abbas AE, Yang L, Zapata R, Schmitz T (2009) Application of decision analysis to milling profit maximization: an introduction. Int J Mater Prod Technol 35(1):64–88
8. Abbas AE, Matheson JE, Bordley RF (2009) Effective utility functions induced by organizational target-based incentives. Journal of Managerial and Decision Economics 30(4):235–251
9. Marples DL (1961) The decisions of engineering design. IRE Trans Eng Manag EM-8 (2):55–71
10. Sage AP (1977) Methodology for large-scale systems. McGraw-Hill, New York
11. Asimow M (1962) Introduction to design. Prentice-Hall, Englewood
12. Siddall JT (1972) Analytical decision-making in engineering design. Prentice-Hall, Englewood Cliffs
13. Starr MK (1963) Product design and decision theory. Prentice-Hall, Englewood Cliffs
14. Rosenstein AB, Rathbone RR, Schneerer WF (1964) Engineering communications. Prentice-Hall, Englewood Cliffs
15. Dixon JR (1966) Design engineering: incentives, analysis, and decision making. McGraw-Hill, New York
16. US News (2016) Best grad schools – industrial/manufacturing/systems engineering 2016. E-source URLhttp://grad-schools.usnews.rankingsandreviews.com/best-graduate-schools/top-engineering-schools/industrial-engineering-rankings
17. Informs(2016) What is operations research. E-source URLhttps://www.informs.org/About-INFORMS/What-is-Operations-Research

Chapter 78
Strengthening Systems Engineering Leadership Curricula Using Competency-Based Assessment

Katherine Duliba and Wilson N. Felder

Abstract The National Academy of Engineering and ABET have emphasized the importance of leadership skills in engineering education. We review engineering education research, including those studies which have assessed engineering curricula using a competency-based approach. Building on this work, we analyze the technical curricula of the Defense Acquisition University (DAU) to assess the extent to which technical leadership is incorporated. Our scope is broader than previous work, which has focused on a single discipline (systems engineering), or a single course (project management), by expanding the scope to five technical disciplines as well as relevant nontechnical courses. Systems engineering leadership competencies are vital in this competency-based assessment. In addition, we extend previous research by developing key competency indicators for technical leadership (several different actions over the junior, mid-level, and senior career stages for each competency), and by incorporating semantic mapping as well as keyword mapping. The results show that at the aggregate level, the DAU curricula include most of the technical leadership competencies, and that at the detailed level (i.e., the key competency indicators), there is the possibility for a higher integration of technical leadership competencies. We suggest that future curriculum assessment work continue to broaden the course scope, deepen the detail by going beyond competencies to key competency indicators, and expand the analysis from keyword mapping to semantic mapping.

Keywords Curriculum assessment • Systems engineering leadership • Technical leadership • Engineering education

K. Duliba (✉) • W.N. Felder
Stevens Institute of Technology, Hoboken, NJ, USA
e-mail: kduliba@stevens.edu; Wilson.Felder@stevens.edu

A.M. Madni et al. (eds.), *Disciplinary Convergence in Systems Engineering Research*, https://doi.org/10.1007/978-3-319-62217-0_78

78.1 Introduction

The National Academy of Engineering (NAE) [1, 2] has encouraged those responsible for educating engineers to augment technical knowledge in science and mathematics with professionalism and leadership, declaring that graduates in 2020 will need professionalism and leadership alongside strong analytical skills. In addition, the NAE identifies creativity, communication, high ethical standards, and agility as attributes of engineers in 2020, attributes which can also be categorized as leadership competencies.

Earlier, in 1997, ABET completed its *Engineering Criteria 2000* (EC2000) project, identifying 11 criteria, or competencies, that engineering baccalaureate graduates should possess [3, 4]. Five of these competencies can be classified as technical, while six competencies can be classified as professional or leadership competencies [5]. The six leadership competencies include multidisciplinary teamwork; ethics; communication; engineering solutions in an economic, environmental, and societal context; lifelong learning; and contemporary issues [4].

Additionally, as organizations seek to recruit engineers possessing leadership skills, developing engineering leadership skills in students during their engineering education is becoming increasingly important. Paul and Falls argue that engineering leadership leads to career success [6].

This paper focuses on an increasingly important aspect of systems engineering education, specifically, systems engineering leadership education. We begin with a review of engineering education studies, including systems engineering education studies, which examine technical leadership in engineering education.

78.2 Background

Shuman et al. conduct an early study examining whether the ABET professional student outcomes criteria can be taught and assessed[5]. They designate six of the ABET student outcomes criteria as “professional” skills, including communication, teamwork, understanding ethics and professionalism, engineering within a global and societal context, lifelong learning, and a knowledge of contemporary issues. To answer the first question, that is, whether the professional skills can be taught, they identify several universities and colleges having courses that address the six ABET professional criteria and describe those courses and programs for each of the six criteria. To address the question of whether the professional skills can be assessed, the researchers conduct an engineering education literature search, and identify several assessment mechanisms, including multisource feedback, peer evaluation methods, project rubrics, tests, attitude measurements, portfolios, competency measures, behavioral observation, external examiner, and performance appraisals. They conclude that the ABET professional skills can definitely be taught and are difficult to assess and that professional skill assessment research is in its early stages.

Kumar and Hsiao also advocate that leadership can be learned through engineering education[7]. They base their argument on ABET's EC2000, the National Academy of Engineers, and a detailed case study. The detailed case study they select is the course, "Geotechnical Engineering in Professional Practice," at Southern Illinois University Carbondale, to illustrate how a university course can successfully contribute to students' learning leadership capabilities in engineering.

Squires and Larson conduct an initial systems engineering curriculum assessment study using a competency-based approach [8]. They select a space industry-based systems engineering competency model consisting of ten overall competencies, which, in turn, consist of 37 systems engineering capabilities. The researchers assess systems engineering curricula using a five-step method. The scope decision, which they view as prior to their five-step method, was to select four core systems engineering courses at one institution, the Stevens Institute of Technology. In the first step of their five-step methodology, they select the curriculum review team, which consists of 13 full-time and adjunct faculty. Next, they select the space industry-based systems engineering competency model. Third, they prioritize the 37 systems engineering capabilities: each of the 13-member review team selects 20 of the 37 capabilities that they view as the most important ones. These results are aggregated for the full review team, resulting in 20 critical systems engineering capabilities. In the fourth step, for each of the critical systems engineering competencies, the team identifies, for each of the four systems engineering courses, two things: (1) if the course addressed the competency (using a three-valued scale [low, medium, and high]) and (2) whether the course should address the competency (also using the same three-valued scale). In the final step, the team identifies action items to address the gaps and opportunities in the systems engineering curriculum.

Applying this methodology, Squires and Larson find that the Fundamentals of Systems and Software Engineering covered 11 of 20 systems engineering capabilities, System Architecture and Design covered 14 capabilities, Systems Integration covered 12 capabilities, and Project Management of Complex Systems covered 6 capabilities. At an aggregate level, all 20 capabilities were covered among the four courses. There were four capabilities that were highly addressed by coursework: create system architectures, define/manage stakeholder expectations, define technical requirements, and logically decompose a system. There were six capabilities that were addressed at a low level by coursework: plan technical effort, manage requirements, manage technical decision analysis process, manage/implement systems engineering, manage team dynamics, and communicate highly effectively. These six capabilities could all be classified as leadership skills.

Özgen et al. conduct a different type of assessment compared to Squires and Larson; they assess engineering students' leadership competencies[9]. In particular, Özgen et al. study chemical engineering students at the Universitat Rovira I Virgili in Spain. The competency model that they use is established on eight essential concepts in the European Foundation for Quality Management (2003) Excellence Model: client orientation, commitment to learning, drive for excellence, integrity,

Table 78.1 Schuhmann et al.'s competencies and keywords

Engineering leadership criteria	Associated keywords
Conceive and design within realistic constraints	Conceive, design
Function within and lead multidisciplinary teams	Team
Identify, formulate and solve engineering problems	Problem-solving, decision, critical thinking, reasoning, and judgment
Understand and demonstrate professional and ethical responsibilities	Ethics
Understand others and communicate effectively	Communicate
Understand economic, environmental, global and societal contexts and impacts	Sustainability, environment, system
Recognize the value of reflection and lifelong learning	Project learning, reflect
Maintain knowledge of contemporary issues	Case study

interpersonal communication, responsiveness to change, results orientation, and teamwork. The researchers assessed leadership competencies using the behavioral event interview and measured leadership effectiveness using 360-degree feedback. They found that the most highly demonstrated leadership competencies were commitment to learning, interpersonal communication, teamwork, and results orientation, while the least demonstrated leadership competencies were integrity, drive for excellence, responsiveness to change, and client orientation.

Similar to Squires and Larson, Schuhmann et al. conduct a study evaluating engineering curricula at ten US civil engineering programs, but these researchers focus on evaluating the engineering leadership content [10].The study selects eight engineering leadership competencies (see Table 78.1), which they term engineering leadership criteria, based on the Bowman and Farr competencies [11], the Gordon-MIT engineering leadership competencies [12], and the ABET student outcomes criteria. They evaluate the curricula using a four-step method. First, they gather the syllabus for the introductory, undergraduate project management course from the civil engineering department, resulting in ten syllabi. Second, the researchers apply keyword analysis to test for the presence or absence of a competency. Third, if a competency was found to be present, the level at which it is present is assigned using a three-valued scale by two researchers (1 = content implicitly present; 2 = content explicitly present; 3 = content is covered in at least one lesson). Fourth, discrepancies between the two researchers' ratings are compared and resolved.

Schuhmann et al.'s results show that two competencies were present the most (20%) (conceive and design within realistic constraints, and understand economic, environmental, global, and societal contexts and impacts), one competency was absent (recognize the value of reflection and lifelong learning), and the remaining five competencies were present in 10% of the sample. The researchers also conducted an institutional coverage analysis and found that two universities included three competencies, three universities included one competency, and five universities did not include any of the competencies. Their study's results are contingent on the project management course selected. Similar to Squires and Larson, they find that communication and multidisciplinary teamwork are addressed at a low level. Overall, Schuhmann et al.'s results indicate that engineering leadership competencies were more absent than present in the project management course in ten US civil engineering programs.

Finally, Palmer et al. review academic engineering leadership development programs (ELDPs), by surveying members of American Society for Engineering Education (ASEE) Engineering Leadership Development Division[13]. Their goal was to identify various innovations in academic engineering leadership development programs. Receiving responses from 30 universities (28 in North America, one in Africa, and one in Europe), they find that several ELDPs have started to incorporate cross-cultural education, and most have applied team-based projects, have a formal mentor program, and lack a corporate sponsor.

In summary, evaluating engineering leadership in engineering curricula is in its early stages. Squires and Larson, examining systems engineering competencies, including some leadership competencies, find that six systems engineering (leadership) capabilities were addressed at a low level in four courses at one academic institution. Schuhmann et al., examining engineering leadership competencies, find that engineering leadership competencies were more absent than present in the project management course in ten US civil engineering programs. With this background, we now turn to the methodology of the current study.

78.3 Methodology

This research is one application of the Technical Leadership Development Framework, a larger study creating a framework to enable individuals and organizations to build technical talent from the junior through to the senior career stages. As this study relies on the competency element of the Technical Leadership Development Framework, we describe the competency element in some detail, before proceeding to the application.

The competencies in the Technical Leadership Development Framework are derived primarily from four sources: Gavito's systems engineering competencies [14], Pyster's systems engineering competencies [15, 16], NASA's systems engineering competencies [17, 18], and NASA's technical executive leadership competencies [19, 20]. The technical leadership competencies consist of 12 technical competencies which have important leadership aspects (for the leaders' technical competencies, see Table 78.2) *and* 12 leadership competencies which enable technical competencies (for the enabling competencies, see Table 78.3). While the technical leadership competencies are partitioned into two groups for ease of understanding, it is important to note that these are all technical leadership competencies (see Fig. 78.1). For example, communication not only enables the technical competencies, but is embedded in most technical leadership competencies, including leaders' technical competencies and enabling competencies. Therefore, as an

Table 78.2 Leaders' technical competencies

Leaders' technical competency	Leaders' technical competency definition
Technical planning	Organizing and scoping the technical work across all the technical phases (from analysis and design, through to development, deployment, and operation)
Technical requirements Definition and analysis	Translating the stakeholder's behavioral and functional needs and expectations into technical statements (including technical problem scope, technical product constraints, and technical requirements).
Logical decomposition	Separating or disintegrating a problem, function, or system into its constituent parts, often into a hierarchical structure
Product verification and validation	Comparing and evaluating the final technical product or system with the initial requirements, specifications, and stakeholders' expectations
Product transition	Deploying the technical product into production, test, operations, and sustainment
Lifecycle	Managing the product movement through the lifecycle, including setting the criteria by which the technical product may be evaluated as it passes from one stage to another
Technical risk management	Identifying, quantifying, and mitigating technical risk, and accepting any residual technical risk
Systems thinking	Seeking holistic explanations and relationships when examining technical problems, and focusing on connections and interfaces among the subsystems in a system
System complexity	Understanding the interfaces within and between systems, and recognizing the potential for emergent behavior due to differences in system components and interfaces
Big picture thinking	Managing the technical aspects external to the system
Abstraction	Identifying and translating a pattern in one domain to a different domain
Paradoxical mindset	Holding opposite views simultaneously to make better decisions

example, communication as a general leadership competency overlaps the technical competency group to form a single intersection, that of technical leadership competencies.

The second step in the methodology is to create *key competency indicators*, that is, different actions that are indicative of the competency being performed at the junior, mid-level, and senior career stages. An example of key competency indicators is provided for the first technical leadership competency, Technical Planning, in Appendix A. The preliminary set of key competency indicators was validated with 44 technical leaders from private and public sector organizations, largely, but not exclusively, drawn from the defense industry (Accenture, Defense Acquisition University, Missile Defense Agency, NASA Marshall Space Flight Center, Navy Strategic Systems, Raytheon Missile Systems, and Sandia National Laboratories). Their revisions and comments were incorporated into the final version. The

Table 78.3 Enabling competencies

Enabling competency	Enabling competency definition
Developing people	Expanding people's ability to do technical work effectively, expanding their ability to lead others effectively, increasing their decision-making capability (with associated trade-offs and judgment calls), helping people understand their career paths and career growth, encouraging people to be good citizens in the workplace, and fostering people's fulfillment from doing their work
Leading people	Guiding, directing, or motivating others in a dignifying and empowering way to further the goals and priorities of the organization
Thinking critically	Using logic and analysis to identify and evaluate the strengths, weaknesses, and implications of different courses of action, as well as analyzing a situation objectively
Building trust	Relating to others in such a way that they believe the leader's intentions and those of the organization
Communicating effectively	Expressing information, meaning and ideas clearly to individuals or groups using verbal, written, and nonverbal skills that help the receiver(s) to understand and retain the message
Enabling competency	Enabling competency definition
Establishing and maintaining stakeholder relationships	Building and sustaining partnerships with other internal or external groups who can impact or are impacted by the technical leader's area
Influencing others	Persuading others to accept a particular view as expressed in an idea, proposal, initiative or decision
Developing strategy and vision	Setting the long-term goals, aligned with organizational goals, evaluating and adopting the courses of action and allocating resources to achieve those goals
Fostering agility	Adapting quickly, learning, responding, and thriving when work tasks, the environment, context, or conditions change; encouraging others to see change as an opportunity and seek betters ways of doing their work
Promoting innovation	Creating, or seeking from others, new or significantly improved products or processes, as well as developing original approaches to handle challenges and opportunities
Possessing government acumen	Making good judgments and managing human, financial, technological, and information resources in a federal, state, or local context, that consists of both federal, state, or local employees and external contractors
Possessing a macro perspective	Delivering solutions within the political, economic, and social aspects, context or landscape

measurement of how well an individual performs a key competency indicator is addressed in a separate part of the Technical Leadership Development Framework.

Having a set of identified technical leadership competencies and key competency indicators, it is now possible to proceed with the assessment. The Defense Acquisition University, the organization having the mandate to train the acquisition

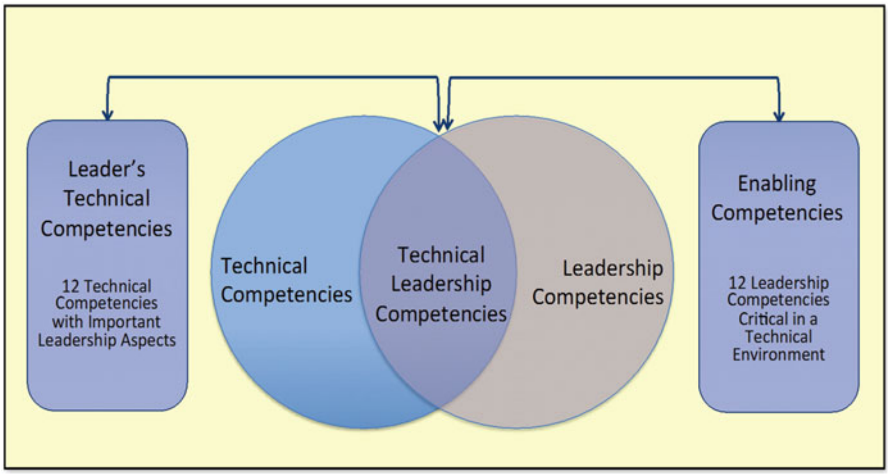

Leader's Technical Competencies
12 Technical Competencies with Important Leadership Aspects
Technical Competencies
Technical Leadership Competencies
Leadership Competencies
Enabling Competencies
12 Leadership Competencies Critical in a Technical Environment

Fig. 78.1 Technical leadership competencies

workforce in the Department of Defense, desires to expand its curriculum from focusing on technical training to also include technical leadership for its technical fields: Engineering; Information Technology; Production, Quality, and Manufacturing; Science and Technology Management; and Test and Evaluation. Within this technical scope, *all* courses are a part of the assessment (19 courses). In addition, 11 courses are deemed to consist of training relevant to some aspects of technical leadership, and so the total number of courses examined is 30. The syllabi for the 30 courses were examined. In addition, the instructor support materials were examined for four important courses that potentially had materials germane to technical leadership. The scope is now set in this third step.

The fourth step is to conduct the gap analysis, consisting of two primary substeps. In the first substep, the relevant objectives from the 30 courses were mapped to the relevant competencies, using both keyword and semantic mapping. Keyword mapping is straightforward, while semantic mapping requires some explanation. Semantic mapping establishes equivalency between two concepts, allowing for synonyms, as well as disallowing the same word with a different meaning. For example, Objective 6 of Engineering 301 states, "Given a system development scenario, the student will evaluate strategies to manage program uncertainty through integration of program metrics and technical measurement with program risk management in accordance with Earned Value Management standards, the DoD Risk Management Guide, and the Defense Acquisition Guidebook." This is a description of technical planning, even though the word is not mentioned. This was confirmed on p. 275 of the ENG 301 Instructor Support Package, "This module is about technical planning, but focuses the part of technical planning that establishes the baselines, measures, and metrics that are used to track progress and manage risk and uncertainty." Consequently, it was necessary to go beyond keyword mapping to semantic mapping.

When a match is found between a course objective and a competency, the second substep is to map the objective to the relevant career stage (junior, mid-level, and senior). This is accomplished by identifying the corresponding verbs between the key competency indicator and the course objective. In this step, close attention is paid to the verbs or verb equivalents in Bloom's taxonomy [21, 22]. Bloom created a hierarchical model of classifying educational objectives, with comprehension (equivalent to understanding) being on the second level, application or "applying" being on the third level, and "evaluating" being on the highest level.

This four-step methodology results in a gap analysis data set that contains, for each of the 24 competencies at three career stages (junior, mid-level, and senior), a set of technical course objectives that address the 24 competencies at the three career stages.

The resulting dataset supports four levels of analysis. The first level of analysis is *competency inclusion*, which is whether the competency (irrespective of career stage or technical area) is included in the curriculum. The second level of analysis is *competency inclusion by area*, which identifies whether the competency is included in the area (Acquisition; Engineering; Information Technology; Production, Quality, and Manufacturing; Science and Technology Management; and Test and Evaluation).The third level of analysis is *competency inclusion by career stage*, which identifies whether the competency at a particular career stage is included in the curriculum. The fourth level of analysis is *competency inclusion by key competency indicator and career stage*, which identifies whether the detailed items comprising the key competency indicator at a particular career stage are included in the curriculum. Applying the four-step methodology together with the four levels of analysis gave the following results.

78.4 Results and Discussion

The first analysis, the competency inclusion analysis, shows that most of the technical competencies and all of the enabler competencies are included in the DAU curriculum. Because the scope was so large (30 courses), it is easier to ensure that a competency is included somewhere in the curriculum. This result is similar to Squires and Larson, who found that, at an aggregate level, all 20 systems engineering capabilities were included in four systems engineering courses. It is also similar to Schuhmann et al.'s aggregate level, where they found that seven of eight engineering leadership competencies were covered among the ten universities they surveyed (Fig. 78.2).

The second level of analysis, the competency inclusion by area analysis, shows that the Engineering and Information Technology curricula cover half of the technical competencies, while the Technical Leadership/Pilot course (TLR-350) covers most of the enabler competencies. This decrease in the coverage level from the aggregate analysis to a more detailed level is also similar to Schuhmann et al.'s study: they found that two universities included three engineering leadership

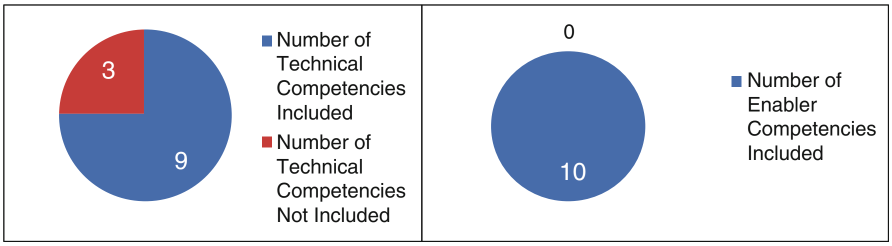


Fig. 78.2 Technical leadership competency inclusion

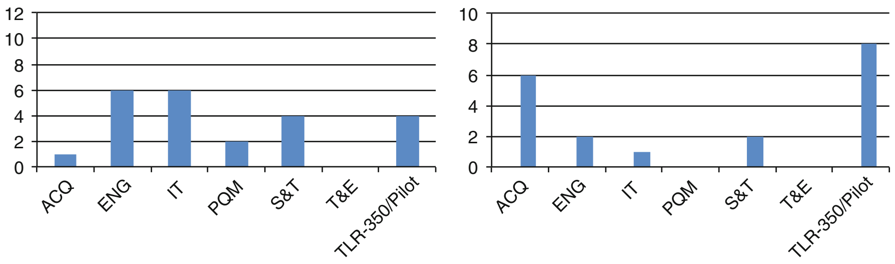


Fig. 78.3 Technical leadership competency inclusion by area

competencies, three universities included a single engineering leadership competency, and five universities did not include any engineering leadership competencies. Squires and Larson also show a decrease at a more detailed level of analysis: one course included 14 systems engineering competencies, a second course 12 systems engineering competencies, a third course 11 systems engineering competencies, and a fourth course 6 systems engineering competencies (Fig. 78.3).

The third analysis, the competency inclusion by career stage analysis, shows that many competencies, both technical and enabler, are included in the curriculum at the junior and mid-level career stages. As the courses are designed for individual contributors and first-level managers, it is not expected that competencies at the senior career stage for second-level managers would be included in the curriculum. As neither Squires and Larson nor Schuhmann et al. incorporate the idea of career stages into their competency model, there is no point of comparison with our analysis (Fig. 78.4).

The fourth level of analysis, *competency inclusion by key competency indicator and career stage*, shows the greatest number of gaps. This, too, is not an unexpected result. As it is the objective of the DAU to expand the technical curriculum from a purely technical curriculum to including technical leadership, the DAU now has the opportunity to take the detailed key competency indicators, by career stage, and identify if and how to change their curriculum to address the key competency indicators. As neither Squires and Larson nor Schuhmann et al. incorporate the idea of key competency indicators into their analysis, there is no point of comparison with our analysis.

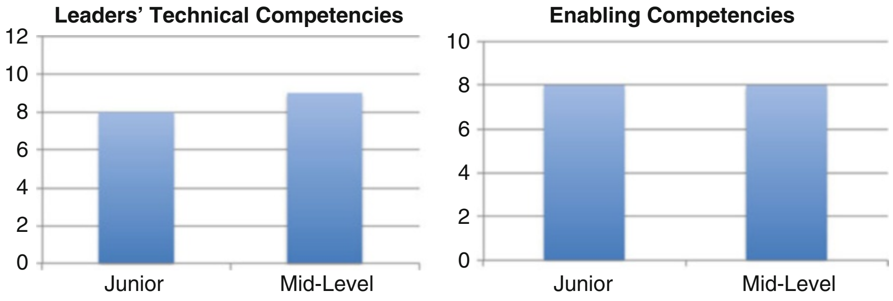


Fig. 78.4 Technical leadership competency inclusion by career stage

78.5 Conclusion

The leadership development of engineers is becoming increasingly important. As academic institutions seek to enhance their curricula to address engineering leadership, we provide a methodology to address this challenge. Based on previous research, our methodology provides two unique contributions: developing key competency indicators for technical leadership and using semantic mapping as well as keyword mapping for the analysis. The results of this analysis are consistent with previous research and provide DAU with the opportunity to expand their technical curriculum with technical leadership.

Acknowledgments This material is based upon work supported, in whole or in part, by the Systems Engineering Research Center (SERC). SERC is a federally funded University Affiliated Research Center managed by Stevens Institute of Technology. We are grateful to all the members of the Technical Leadership Development Framework team, including members of participating universities, sponsor members, and subject matter experts. In particular, we would like to thank David Pearson, Defense Acquisition University, and Aileen Sedmak, Office of the Deputy Assistant Secretary of Defense for Systems Engineering, for their contribution to the Framework, and this specific study.

Appendix A: Key Competency Indicators for the Technical Planning Competency

Career stage		
Junior	Mid-level	Senior
Competency class 1:leaders' technical competencies		
1.1 Technical planning		
Develops technical plan for a specialized item, under the coaching of mid-level leaders; Relays convincing, clear and	Develops and details out the technical plan for a system to fit into the overall technical plan for a large (or complex) system;	Develops overall technical plans, for a large (or complex) system, that: Support the strategy, vision, mission and long range goals

(continued)

relevant information from the specialized item technical plan to mid- level leaders	Reviews and approves technical plans for specialized items developed by j unior-level leaders; Guides, directs and coaches junior- level leaders to detail out a technical plan for a technical component; Provides clear direction from the system-level technical plans down the hierarchical levels to junior-level leaders; Reiavs convincing, clear and, relevant information from the system-level technical plan to senior-level leaders; Coordinates the system-level technical plan and obtains consensus among peer internal suborganizations (both technical and non-technical);	(which recognize needs) of the organization or enterprise; Provide direction to mid-level leaders; Is aligned with and supports the plans and objectives of peerorganizations, both technical and non-technical; Reflect the technical impact of the superior organization or enterprise's strategies and missions. Reviews and approves technical plans for a product or a system developed by subordinate suborganizations; Guides, directs and coaches mid-level leaders to detail out the overall technical plan for a large (or complex) system into the appropriate detailed plans; Provides clear direction from the technical plans down the hierarchical levels to subordinate suborganizations and their leaders; Relays convincing, clear and, relevant information from technical planning up the hierarchical levels in the enterprise; Coordinates the technical plan and obtains consensus (using influencing and negotiation skills) among peer internal suborganizations (both technical and non-technical); Represents and communicates the overall technical plan in the larger technical and non-technical community; Communicates clear, relevant technical plan information to external organizations, including partners in other agencies, industry, academia and perhaps internationally, raising awareness and identifying potential areas of agreement and disagreement among external organizations

References

1. NAE (2004) The engineer of 2020:visions of engineering in the new century. The National Academies Press, Washington, DC
2. NAE (2005) Educating the engineer of 2020:adapting engineering education to the new century. The National Academies Press, Washington, DC
3. ABET (2002) Criteria for accrediting engineering programs:effective for evaluations during the 2002–2003 accreditation cycle. ABET, Inc, Baltimore
4. ABET (2014) Criteria for accrediting engineering programs:effective for reviews during the 2015–2016 accreditation cycle. Baltimore, ABET
5. Shuman LJ, Besterfield-Sacre M, McGourty J (2005) The abet "professional skills"–can they be taught? Can they be assessed? J Eng Educ 94(1):41–55
6. Paul R, Falls LC (2015) Mapping career success competencies to engineering leadership capabilities. IEEE frontiers in education conference, IEEE, El Paso
7. Kumar S, Hsiao JK (2007) Engineers learn "soft skills the hard way":planting a seed of leadership in engineering classes. Leadersh Manag Eng 7(1):18–23
8. Squires A, Larson W (2009) Improving systems engineering curriculum using a competency-based assessment approach. IntJIntellDefSupport Syst 2(3):184–201
9. Özgen S et al (2013) Assessment of engineering students' leadership competencies. Leadersh Manag Eng 13(2):65–75
10. Schuhmann RJ, Magarian JN, Huttner-Loan E (2014) A method for assessing engineering leadership content in the engineering curriculum: a first look at civil engineering project management courses. ASEE annual conference & exposition, Indianapolis
11. Bowman BA, Farr JV (2000) Embedding leadership in civil engineering education. J Prof Issues Eng Educ Pract 126(1):16–20
12. Bernard M(2011) Gordon-MIT engineering leadership program. Capabilities of effective engineering leaders, version 3.6, Boston
13. Palmer JCet al (2016) Leading the way:a review of engineering leadership development programs. ASEE annual conference &exposition, New Orleans
14. Gavito V et al (2010) Technical leadership development program. Systems Engineering Research Center, Stevens Institute of Technology, Hoboken
15. Pyster A et al (2014) Atlas:the theory of effective systems engineers, version 0.25. Systems Engineering Research Center, Stevens Institute of Technology, Hoboken
16. Pyster A et al (2015) Atlas:the theory of effective systems engineers, version 0.5. Systems Engineering Research Center, Stevens Institute of Technology, Hoboken
17. Williams C, Derro M-E (2008) Nasa systems engineering behavior study,NASA
18. NASA (2008) Nasa's systems engineering competencies,NASA
19. Morris LE, Williams CR (2012) A behavioral framework for highly effective technical executives. Team PerformManag: An InterJ 18(3/4):210–230
20. Williams C et al (2010) Executive leadership at nasa: a behavioral framework.National Aeronautics and Space Administration. http://www.nasa.gov/offices/oce/appel/seldp/resources/exec_leadership.html
21. Bloom BS et al (1956) Taxonomy of educational objectives, handbook I:the cognitive domain. Longman, New York
22. Anderson LW et al (2001) A taxonomy for learning, teaching, and assessing:a revision of bloom's taxonomy of educational objectives. Pearson, New York

Chapter 79
Integrating Systems Engineering Students in Capstones: A Multispectrum Characterization of Interdisciplinary Capstones

Cory A. Cooper, Jeremy J. Homan, and Brian E. Tidball

Abstract The use of the engineering capstone experience is ubiquitous in engineering education. The development of undergraduate systems engineers (SE) presents a unique challenge to both prepare them in a standardized and traceable way and purposefully place them into domain-centric engineering projects that allow them to develop their experience base (i.e., the leg of the "T"-shaped SE). In order to better understand the various characteristics of the United States Air Force Academy's (USAFA) enterprise of over 30 projects in seven hosting departments, this research uses established frameworks to search for correlations to good capstone learning objective performance. Two years of capstone student self-assessment data was used to inform the authors in addition to qualitative faculty observations. These two primary sources of data were then analyzed for intersecting conclusions. Four common conclusions were found: (1) capstone projects should have a mix of internal/external funds, (2) the project should have a reasonable mix of flexibility and structure in the degree of constraints, (3) external customers should be involved for each project, and (4) team sizes should nominally include 3–10 students. Limitations of this research and the conclusions are also discussed.

Keywords Capstone • Framework • Undergraduate • Characteristics • Spectrum • Systems engineering • Rubric

C.A. Cooper (✉) • J.J. Homan • B.E. Tidball
United States Air Force Academy, Colorado Springs, CO, USA
e-mail: cory.cooper@usafa.edu; jeremy.homan@usafa.edu; brian.tidball@usafa.edu

79.1 Introduction

Whether designing an engineering capstone experience for undergraduate systems engineers or planning and staffing complex, multidisciplinary engineering projects in government and industry, there are certain project characteristics that are taken into account. Experienced technical leaders know which characteristics should be adjusted in order to provide the largest chance for project success. Many times these heuristics are dependent on a leader's unique experiences or the host organization's culture. At the United States Air Force Academy (USAFA), systems engineering students participate in a year-long senior engineering design capstone experience. Ultimately, they graduate with an Accreditation Board for Engineering and Technology (ABET)-accredited systems engineering degree. ABET's focus on assessment-supported and institution-aligned Program Educational Outcomes (PEOs) enforces a strong traceability mindset with respect to flow-down of program outcomes to learning objectives in each supporting course in the curriculum. The capstone experience, as a culminating piece of the SE curriculum's broad aims, can be described by the student learning objectives it hopes to achieve:

- Understand and implement rigorous systems engineering practices.
- Critically analyze and trade off program requirements and constraints (cost, schedule, and performance) to develop realistic system design options.
- Demonstrate independent learning by researching and assessing specific issues of system performance and applying them to individual team tasks.
- Demonstrate an ability to work effectively as a member of a Systems Program Office team, in both leader and follower roles, by understanding program goals and objectives; identifying problems, analyzing alternatives, and implementing solutions; diligently tracking and documenting decisions and analytical results; and successfully completing program milestones.

These outcomes are supported by a capstone enterprise at USAFA that integrates 7 separate engineering departments and over 30 distinct projects each year. The systems engineering program at USAFA purposefully places each of its 80+ students each year into the 30+ projects in a distributed model. This approach aligns with the SE program's belief that the central tenets, skills, and processes taught in the SE curriculum are domain-independent and can be applied to any domain. With this, it is also held that the practice of SE within these other domains is one of the best ways to hone SE knowledge and overcome the lack of experience seen in undergraduate SE education. This approach to distributing SE students into domain-hosting departments comes with the risk of not fully grasping how the students are best able to achieve the SE learning objectives listed above. For this reason, the authors and other faculty in USAFA's SE program have undertaken a research project over the past few years to understand which characteristics of the 30+ projects best support the SE students and the ability of the program to produce well-prepared SE graduates. Some projects have small teams (3–4 members), are ill-defined, internally funded, and follow agile methodologies in system development. Other projects are multiyear; have large teams (25–30 members), with multiple advisors and external funding; and follow the DoD acquisition process. How can faculty compare the two experiences for embedded SE students in each, and where should the capstone enterprise move to in order to support our SE capstone learning objectives?

79.2 Related Research

Previous research by the authors has studied related research and the lack of a more holistic, multispectrum approach to understanding capstone project characteristics [1–3]. These referenced publications present a more thorough review of the related research than will be presented here. Most related prior research focused on assessment or one to two specific capstone characteristics. It was observed that previous research did not study more than two characteristics at a time and lacked a holistic look at other factors that could be considered in capstone project design. Prior related research in the area of assessment looked at methods for assessing capstone experiences with respect to the student learning outcomes [3, 4]. Other related research studied the singular or comparison of characteristic pairs in project development (e.g., single to multidisciplinary teams [5, 6], small vs. large groups [7], short vs. long duration projects [3]), for various stakeholders. There are a few related frameworks that have been used to understand capstone projects (e.g., quality function deployment (QFD) approach [8]; a project elements approach (student preparation, project selection, and instructor mentorship) [9]; an uncertainty, complexity, and pace (UCP) model [5]; a four essential elements approach (real-world problems, using a total design process, closing engineering competency gaps, and integration of technical skills) [10]; and a “hard” vs. “soft” projects dimensional approach [11]). However, in each of these frameworks and comparisons, a direct application to the questions of the current research is incomplete.

79.3 Approach

Continuing from previous research by the authors, the objective of the current research is to understand the following research questions as part of a comprehensive study on the classification of capstone courses. Observations are based on the correlation between characteristics and students’ attainment of capstone learning objectives.

79.3.1 Research Questions

The following questions represent three major phases of investigation that this research seeks to explore.

1. Is it possible to establish a common framework for characterizing the full breadth of capstone experiences at USAFA? A framework of 19 capstone characteristics has been established and implemented to support further study [1].
2. Is there a relationship between capstone characteristics and student performance (with respect to capstone course student outcomes)? Initial results have been collected, analyzed, and reported [2, 3]. This paper presents further results to refine continued observations.
3. How can the established framework and observed relationships be used to improve future capstone offerings and placement of students in capstones to increase the achievement of Course Student Outcomes? Analysis of results and the sharing with the broader community continues [3, 12, 13].

Therefore, this paper seeks to answer the question, "What correlations are observed between project characteristics and capstone learning outcomes following a 2-year study of undergraduate systems engineering students on a diverse set of engineering capstones?"

79.3.2 Research Approach

This research uses a constructivist approach [14] in that it has established a novel framework for understanding the characteristics of capstone projects and forms the basis for further study of the impact of those characteristics. The multispectrum framework established by the authors [1] created rubrics for the following 19 capstone project characteristics:

1. Funding source (e.g., external to internal)
2. Degree of constraints (e.g., open-ended to highly constrained)
3. Starting point for requirements refinement (e.g., ill-defined to existing requirements)
4. Agility of design process (e.g., stepwise to as-needed)
5. Diversity of team member major/skillset (e.g., multidisciplinary to homogeneous)
6. Scope of programmatic concern (e.g., project team to program office)
7. Reflection of DoD development process (e.g., DoD acquisition process to rapid innovation)
8. Customer involvement (e.g., internal to external)
9. Team size (e.g., small to large)

10. Novelty of project (e.g., original problem/solution to existing project framework)
11. Formality (e.g., formal to informal)
12. Potential for publishing work (e.g., expected to exception)
13. System level (e.g., consumer-level product design to system of systems)
14. Course maturity (e.g., new construct to mature offering)
15. Knowledge use (e.g., new knowledge required to application of previously learned material)
16. Other faculty involvement (e.g., single instructor to team of instructors)
17. Team selection (e.g., volunteer filled to directed)
18. Competitive (e.g., sole-source to competition)
19. Mission linkage (e.g., military need to general application)

Several characteristics are of unique interest to USAFA as a US Government organization. For example, reflection of the DoD development process or linkage to a military need is directly useful for study at USAFA. Table 79.1 shows an example rubric for one of the 19 characteristics, "agility of design process."

Rubrics for the four capstone learning outcomes were also developed [2]. A combination of the 19 characteristic rubrics and the learning objective rubrics were then provided for self-assessment by students in two separate years of capstones, 2015 and 2016. Relationships between the characteristics and learning objective attainment were explored through both quantitative (Pearson product-moment correlation calculations) and qualitative (observations by SE faculty central to all 30+ projects) methods.

For the quantitative analysis, scores were correlated for each of the 19 capstone characteristics and the 4 learning objectives (i.e., a 19 x 4 matrix of r-values). Calculated r-values range from negative one to positive one. Large positive values indicate a strong positive correlation, and large negative values indicate strong negative correlation. Due to an arbitrary assignment of the number 1–5 to the characteristic rubrics, both the strong positive and strong negative correlations are of interest. The sign simply will indicate the end of the spectrum that is correlated. Small values at, or around, zero indicate that a weak or no correlation is present.

For the qualitative analysis, the authors provided independent observations about the capstone enterprise from their direct and overseeing roles within the USAFA capstone enterprise. These independent observations were then organized by general categories of the capstone characteristics. Observations that were common to multiple authors were summarized and are listed following a brief description of the general characteristics categories.

Table 79.1 Example rubric characterizing the "agility of design process" spectrum (capstone characteristic #4)

Stepwise				As-needed
Syllabus of design topics established and followed rigorously	Syllabus of design topics established, but can be adjusted in certain cases	Syllabus of design topics/tools is mostly defined, but adapted regularly to the project progression	Basic design topics/tools are presented and then augmented as needed	Design topics/tools introduced only as needed

79.4 Results

This section presents the results of the student self-assessment surveys and qualitative observations of the faculty over the past two academic years. The self-assessment survey results are similar in format to previous reporting for ease of comparison. The qualitative observations of the authors provide a lens of experience that may be masked by simply looking at the student self-assessment data alone.

79.4.1 Quantitative Results

Previous research [2] presented data from an initial application of the multispectrum rubrics from the class of 2015 at USAFA. The following tables and correlations are presented in a similar format for ease of comparison to that research. It was important to continue gathering data in a standardized manner to be able to draw conclusions with less potential for temporal factors that may have existed only for the earlier class of students.

Students from two class years were surveyed with the rubrics presented in previous research [1, 2]. From this pool of 155 from the classes of 2015 and 2016, 93 students responded (60%). The characteristic rubrics were assigned an arbitrary scale from 1 to 5 from left to right. The learning objective rubrics were assigned an increasing quality scale from 1 to 5 from left to right (i.e., 1 is substandard, 5 is exemplary). These scales were necessary to allow quantitative correlation analysis.

The collected assessment data was averaged by hosting departments to reduce the impact that the number of respondents had on the overall data averages and correlations. This step was necessary due to some departments returning a higher proportion of feedback forms than in other departments. Grouping of data by department is reasonable given the high cultural alignment of projects within each domain. The response data for capstone characteristics was then compared to response data for the learning objectives to observe any relatively strong correlations using a Pearson product-moment correlation calculation. These r-value coefficients can be seen in Table 79.2. Because the characteristic rubrics were assigned a numeric scale arbitrarily without presumption of value direction, both strong positive and strong negative correlations are of equal interest to the study.

- Capstone characteristics that exhibit the *highest relative correlation* to learning outcomes:
  - For LO1 (*Understand and implement rigorous systems engineering practices*)
    - "agility of the design process (4)" – as-needed preferred
    - "reflection of the DoD development process (7)" – DoD process preferred

Table 79.2 Pearson correlation coefficients (r) between learning outcome (LO) attainment and capstone characteristics, combined 2015–2016 data. *Color shading* is included to highlight relative strong (*dark*) and weak (*light*) correlations

	Capstone Characteristics																		
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
LO1	-0.31	0.07	0.31	0.68	-0.15	0.45	-0.77	0.63	0.61	0.38	0.42	-0.42	0.21	0.50	-0.46	0.21	-0.32	0.15	-0.32
LO2	-0.12	0.06	0.50	0.60	-0.19	0.55	-0.69	0.59	0.75	0.62	0.37	-0.24	0.39	0.58	-0.31	0.32	-0.27	0.25	-0.32
LO3	-0.01	-0.31	0.26	0.80	0.46	0.49	-0.76	0.65	0.38	0.19	0.82	-0.31	0.03	0.11	-0.38	-0.09	-0.78	-0.44	-0.16
LO4	-0.17	-0.01	0.08	0.11	0.49	-0.29	-0.42	0.07	-0.02	-0.14	0.31	-0.46	-0.23	-0.09	-0.13	0.01	-0.57	-0.57	-0.08

From the 2015 and 2016 correlation data, the following observations were made. The observations are organized by learning objectives and a characteristic "direction preference" is indicated based on alignment with the sign of the correlation value

    - "customer involvement (8)" – external customer preferred

  - For LO2 (*Critically analyze and trade off program requirements and constraints (cost, schedule, and performance) to develop realistic system design options*)
    - "reflection of the DoD development process (7)" – DoD process preferred
    - "team size (9)" – large team size preferred
  - For LO3 (*Demonstrate independent learning by researching and assessing specific issues of system performance and applying them to individual team tasks*)
    - "agility of the design process (4)" – as-needed preferred
    - "reflection of the DoD development process (7)" – DoD process preferred
    - "process formality (11)" – informal process preferred
    - "team selection (17)" – volunteer preferred
  - For LO4 (*Demonstrate an ability to work effectively as a member of a Systems Program Office team, in both leader and follower roles*)
    - "skill diversity of team (5)" – homogeneous preferred
    - "team selection (17)" – volunteer preferred
    - "competitive (18)" – sole-source preferred

- Capstone characteristics that seem to have *no correlation* to the learning outcomes:
  - For LO1
    - "degree of constraints (2)"
    - "skill diversity of team (5)"
    - "competitive (18)"
  - For LO2

    - “funding source (1)”
    - “degree of constraints (2)”

- For LO3

    - “funding source (1)”
    - “system level (13)”
    - “other faculty involvement (16)”

- For LO4

    - “degree of constraints (2)”
    - “team size (9)”
    - “other faculty involvement (16)”

The above data was aggregated to observe which characteristics had the greatest impact on the capstone as a whole. This was done by averaging the absolute values of the r-values in each characteristics column to equally consider extremes on either end of characteristic spectrums as the original rubric scales were arbitrarily assigned. Table 79.3 displays these correlation coefficients.

The top three characteristics that have a high average correlated effect on the learning outcomes are “agility of the design process (4)” – as-needed preferred, “reflection of the DoD development process (7)” – DoD process preferred, “process formality (11)” – informal process preferred, and “team selection (17)” – volunteer preferred. Also, “funding source (1),” “degree of constraints (2),” and “other faculty involvement (16)” exhibit the lowest three average correlations with learning outcome attainment

79.4.2 Qualitative Observations

In addition to the student rubric responses, the authors made qualitative observations. Each of the three authors performs major roles across the 30+ capstone projects each year. Their involvement spans seven engineering departments and includes leading individual projects, providing specific domain expertise, and mentoring faculty and students on systems engineering processes and tools. From this experience base, the authors independently documented observations regarding four general categories of characteristics that help characterize and differentiate the projects: customer, process, team, and scope/goal.

Characteristic categories alignment to rubric characteristics:

- Customer: 1, 3, 8, 19
- Process: 2, 3, 4, 7, 11, 14, 16
- Team: 5, 9, 17
- Scope: 6, 10, 12, 13, 15, 18, 19

Table 79.3 Average of absolute values of Pearson correlation coefficients (r) for each capstone characteristic, combined 2015–2016 data

	Capstone characteristics																		
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
AVG|r|	0.154	0.111	0.285	0.546	0.322	0.444	0.660	0.485	0.441	0.336	0.482	0.359	0.213	0.320	0.321	0.156	0.486	0.351	0.221

The following observations attempt to link project characteristics to the feasibility of administering capstone projects while meeting the systems engineering learning outcomes. Observations listed reflect the intersection of independent observations of the authors when they considered each of the four characteristics categories.

79.4.2.1 Customer-Based Characteristics

Most projects have an external customer, though involvement can vary dramatically. Customers typically provide the project objective (user needs), a starting point (in the system lifecycle: novel or enhancement), funding (and or technology), and expertise.

- Customer engagement seems to be the driving characteristic of success. Providing a clear objective, willing access to technical and user expertise, and enthusiasm about student ideas and solutions supports an engaged and motivated student team.
- A combination of customer and department funding seems to work well in avoiding real-world constraints in purchasing parts and supplies, minimizing faculty oversight, and avoiding uncontrollable project slowdowns.
- Customers with preselected solutions or excessive change in requirements, while realistic, frustrate students and create additional work for faculty that detracts from the overall experience. These real-world project issues are a uniquely interesting issue for the systems engineering students, but tend to pull the project too far from the understanding that it is still an undergraduate level of project work.

79.4.2.2 Process-Based Characteristics

The capstone design process at USAFA generally mimics the milestones of the DoD acquisition process, scaled for the academic year. Departments and faculty vary in the deliverables they emphasize and the degree of freedom given to students. Projects with extensive computer programming utilize a rapid iteration or agile project management methodology (e.g., Scrum process).

- A process that balances recommended milestones and deliverables, with student-driven freedom to align schedule and deliverables with their project scope seems a good and necessary approach.
- When process and deliverables are highly dictated, faculty oversight and grading is simplified, but often completed as “check boxes” to satisfy the grading with little to no value added to the project.
- When students are given high process freedom, they often produce something interesting, but struggle with meeting schedule and or technical objectives.

79.4.2.3 Team-Based Characteristics

The typical team size is 6 students, but varies from 3 to 12 depending on project scope and complexity. Six-member student teams usually include three to four students from the host discipline and two to three SE students, while larger teams incorporate a diverse mix of engineering disciplines, and divide into interdisciplinary subsystems.

- The five- to six-member student team size seems to have independently developed in each department, allowing for a technically complex problem that is interesting and useful while requiring the effort, expertise, and coordination of a team.
- Interdisciplinary teams take longer to clarify member roles and have an overall slower start, but seem to produce more technically sound solutions that more fully meet customer requirements.
- Larger teams struggle from added coordination between subsystem teams and large-scale integration. This more directly conflicts with competing requirements on the students' time (e.g., other courses and nonacademic activities).
- Smaller teams usually work toward less complex projects, which results in feeling more like an assignment than a significant technical contribution. However, there is evidence of technically complex projects in small computer science projects that use agile development and where hardware manufacturing is not necessary.

79.4.2.4 Project Scope-Based Characteristics

The capstone project scope and ultimately the end deliverable is often a negotiation between the customer, faculty, and students. Emphasis is placed on ensuring students experience the product lifecycle (requirements analysis through construction of a functional prototype). Faculty experience is used to scale technical complexity to challenge students, meet customer needs, and achieve what is feasible during and academic year.

- Scope or project objectives that originate from real customer needs and are tailored by faculty experience seem to coherently meet the process, technical, and educational outcomes of the capstone experience.
- Larger scope requires dedication of students to managerial roles, applied to technical integration. This can be positive, but usually only involves coordination and communication activities (i.e., lack of engineering rigor in those students' contribution). Large projects also require added faculty oversight and planning, and usually have a momentum that does not allow for consideration of failure as a system developmental methodology.
- Highly dictated or constrained scope becomes a "build to spec" project, limiting student learning and experience. Allowing for student requirements definition and selection creates student buy-in and improved quality of the process and final outcome.

79.5 Discussion

The results presented in the previous section will be discussed here in two ways. First, the quantitative and qualitative results will be explored for overlapping conclusions. Second, limitations on this research will be discussed in order to highlight applicability and potential for further research.

79.5.1 Analysis of Results

With 2 years of collected data, the quantitative results are becoming more representative of a possible steady state of conclusions from self-assessment data. However, as will be discussed in the next subsection, there are limitations to this self-assessment data. An appropriate lens of experienced faculty is necessary to observe more supported correlations of performance. An alternate risk also exists in having faculty provide observations, in that bias may exist in the view of the "ideal" capstone model. For these reasons, conclusions presented here represent only the intersection of conclusions that can be drawn from both sources, quantitative and qualitative.

The first characteristic that seems to be supported well by both the correlation data and faculty comments is "funding source (1)." Correlation data indicated that good learning objective performance was not correlated with this characteristic; however, a more appropriate description would be that it may be correlated more closely with the middle of the characteristic's spectrum (i.e., funding should be a shared mix of internal and external sources). This shared mix of funding ensures that there is the ability to travel for stakeholder engagement and purchase prototyping materials, but it also ensures the committed interest by each party and perception of moderate stakes by the student team.

The second characteristic that seems to be well supported is the "degree of constraints (2)." Similar to the "funding source (1)," good learning objective performance seems to be more closely with the middle of the characteristic's spectrum (i.e., some aspects of the project are defined, but there is flexibility in the formation of the project topics and deliverables). This reasonable approach to the project constraints allows enough freedom to be innovative and responsive to real-world activities, but gives enough structure for students in their first experience with complex engineering processes.

A third characteristic that seems well supported is "customer involvement (8)." For this characteristic, the correlation data and the faculty remarks indicate a good learning objective performance aligns with having external customers involved. Similar to the discussion on funding source, having an external customer increases a perception of having a stake in the project and an unwillingness of the student team to let an external actor down. The external customer brings a fresh perspective to the engineering design process that may be welcomed when student become comfortable with faculty in simply a classroom setting.

A final characteristic that is supported by both sources is "team size (9)." Similar to the first two characteristics described above, good learning objective performance appears aligned with the middle of this rubric (i.e., 3–10 members). Either end of this characteristic's spectrum can create issues with a project being either too simple (i.e., individuals or pairs) to adequately prepare systems engineers or too large (i.e., teams of 11 or more) to allow each team member to equally contribute in multiple tasks and rigorous engineering methodologies.

While other characteristics did not have as clear an overlap as these four characteristics, there are several other areas of interest that should be considered when designing capstone teams.

79.5.2 Limitations of Research

79.5.2.1 Response Rate and Domain/Discipline Distribution

Several limitations regarding this research exist. The following are some of the most relevant to the results. The first limitation observed is the moderate response rate of students. Combining both academic years, there were 155 students enrolled in the various projects and 93 (60%) elected to respond to the surveys. Of these responses there is also an unequal domain/discipline response distribution. Two of eight SE depth areas reported half as many responses as the other six. However, of the remaining six disciplines, the responses are relatively proportionate. Although not ideal, this response rate and distribution shows no indication for unfair bias within the data. Nothing in the observed data would lead one to conclude that the remaining 40% would significantly alter the reported results.

79.5.2.2 Confusion with Capstone Characterizations

The survey results include the assumption that respondents would answer accurately and understand the content within the capstone characterization questions. Some respondents could, for reasons such as apathy or confusion within their project, respond inaccurately. For example, it is conceivable that a student could view his or her project as reflecting the DoD acquisition process when it is in fact reflecting rapid innovation, merely for not understanding the difference. Although all students were instructed to answer the survey questions to the best of their knowledge, they were not explicitly recommended for or against guessing when a student might be unfamiliar with the content. While this presents a possibility for misinterpreted responses, the risk to significantly skewing the data was assessed by the authors as unlikely.

79.5.2.3 Learning Outcomes Self-Assessment and Bias

Similar to the capstone characterization assumption, the authors believe that students were capable of determining the attainment of learning outcomes related to their capstone. This assumption provides one of the greatest risks in the research as self-assessment is inherently difficult and the students are likely biased toward their own success. This initial research does not propose causality, but instead studies the relationship between the students' perceived characterization of each capstone and the learning outcomes they perceive as being met. Learning outcome attainment could be severely biased by students that for many reasons did or did not enjoy their capstone experience. For example, a student could have been placed on a capstone project with a domain that is of very little interest to that student. It is possible that this student would report learning outcome attainment as low due to the disinterest in the material, when in reality the student learned a fair amount. A similar example exists in how a positive or negative student-instructor relationship could bias the student's perceived learning outcome attainment.

Continuing the discussion on limitations related to self-assessment, the data analysis indicates that the learning outcome self-assessments tended to lean more toward the "proficient" end of the spectrum. Additionally, the majority of the capstone projects were reported as having similar characteristics (i.e. team size, funding source, etc.). These correlation values tend to indicate that exemplary learning outcome attainment is associated with what the majority of the surveyed project characteristics were reported as. This limitation could be attributed to many factors to include a student's bias toward their own success (i.e., "my capstone looks like this, and of course I was successful at learning").

79.6 Future Research and Conclusion

79.6.1 Future Research

Continued research is needed to fully understand the best mix of capstone characteristics for exemplary learning outcome attainment, and explore possible characteristics missing in an observed set of capstone projects. It is feasible that the best mix of characteristics is being masked by the tyranny of the "normal" capstone for USAFA. Furthermore, this research only proposes that a correlation between capstone characteristics and learning outcomes exists, but does not propose that only the researched characteristics matter, or matter most. Instead, it is likely that other factors to include student aptitude, attitude, personality, or similar for the faculty advisors could have a much more significant impact on learning outcome attainment. Further research could explore these other factors as they compare to capstone characteristics.

79.6.2 Conclusion

Careful construction of engineering capstone experience is necessary when balancing the need to produce systems engineering undergraduates in a standardized manner with the need for purposeful placement of SE students across a 30 project, 7 domain capstone enterprise. The present research has sought to better understand the many characteristics that make capstone projects unique and how they affect capstone learning objective performance. Through the collection of student self-assessment data and faculty qualitative observations, a few conclusions are supported. The capstone director seeking to achieve good SE learning objectives should consider shaping their capstone projects based on the following guidelines: (1) the project should have a mix of internal/external funds, (2) the project should have a reasonable mix of flexibility and structure in the degree of constraints used, (3) external customers should be involved for each project, and (4) team sizes should nominally include 3–10 students. These conclusions have certain limitations and care should be taken to review the full listing of results and consider other factors as capstones are adapted for optimal learning performance.

Acknowledgments The authors would like to thank the Scholarship of Teaching and Learning (SoTL) center at USAFA for their shared interest and assistance in this research. The authors would also like to thank the many capstone mentors and instructors that have collaborated on the proposed framework through discussions and explanations of their specific capstones.

References

1. Cooper CA, Fulton JM, Homan JJ (2015). A multi-spectrum framework for characterizing interdisciplinary capstone design experiences. In: Proceedings of the 13th annual conference on systems engineering research (CSER), Stevens Institute, Hoboken
2. Cooper CA, Fulton JM, Homan JJ (2016) Initial application of a multi-spectrum characterization framework for interdisciplinary capstone design experiences. In: Proceedings of the 14th annual conference on systems engineering research (CSER), Huntsville
3. Homan JJ (2016) Initial application of a framework for improving the capstone educational experience at the US Air Force Academy. In: Presentation to the military operations research society (MORS) 84th symposium, Quanitco
4. Zable J (2010) Guest editorial: 2007 national capstone design conference. Adv Eng Edu 2 (1):1–4
5. McKenzie LJ, Trevisan MS, Davis DC, Beyerlein SW (2004) Capstone design courses and assessment: a national study. In: Proceedings of the 2004 American society of engineering education annual conference and exposition, Salt Lake City
6. Meyer DG (2005) Capstone design outcome assessment: instruments for quantitative evaluation. In: Proceedings of the 35th ASEE/IEEE frontiers in education conference, Indianapolis
7. Dvir D, Shenhar AJ, Alkaher S (2003) From a single discipline product to a multidisciplinary system: adapting the right style to teh right project. Syst Eng 6(3):123–134
8. Rover DFP (1997) Cross functional teaming in a capstone engineering design course. In: Proceedings of the 27th annual frontiers in education conference, Pittsburgh

9. Griffin P, Griffin SO, Llewellyn DC (2004) The impact of group size and project duration on capstone design. J Eng Educ 93(3):185–193
10. Noble JS (1998) A approach for engineering curricukum integration in capstone design courses. Int J Eng Educ 14(3):197–203
11. Marin JA, Armstrong JE, Kays JL (1999) Elements of an optimal capstone design experience. J Eng Educ 88(1):19–22
12. Cooper CA, Homan JJ, Fulton J (2015) Characterization of capstone design experiences. In: Presentation to the 7th annual scholarship of teaching and learning (SoTL) fourm, USAFA
13. Cooper CA, Homan JJ (2016) Characterization of capstone design experiences. In: Presentation to the 8th annual scholarship of teaching and learning (SoTL) forum, USAFA
14. Ferris TL (2009) On ethe methods of research for systems engineering. In: Proceedings of the 7th annual conference on systems engineering research, Loughborough
15. Crawford L, Pollack J (2004) Hard and soft projects: a framework for analysis. Int J Proj Manag 22(8):645–653

Chapter 80
SEEA: Accelerated Learning and Learning Assessment for Systems Engineering Education

Peizhu Zhang, Jon Wade, Richard Turner, Douglas Bodner, and Dale Thomas

Abstract The Experience Accelerator is a new approach to developing the systems engineering and technical leadership workforce, aimed at accelerating experience assimilation through immersive, simulated learning situations where learners solve realistic problems. A prototype technology infrastructure and experience content has been developed, piloted, and evaluated. This paper discusses the use of the technology in the systems engineering education domain. An evaluation of the learning potential is presented utilizing the data collected from a pilot application of the prototype in an undergraduate course on project management. Finally, a summary is provided with a description of future work.

Keywords Systems engineering education • Learning assessment • Accelerated learning

80.1 Introduction

Systems engineering (SE) and technical leadership are multidisciplinary practices that are as much an art as a science. While a traditional model of education can teach the fundamental body of knowledge, it is not until the knowledge is put into real-world practice that a systems engineer can develop the required insights and wisdom to become proficient. Due to the exponential advancement of technology, rapidly evolving needs, and increasing systems complexity, it is even more

P. Zhang (✉) • J. Wade • R. Turner
Stevens Institute of Technology, Hoboken, NJ, USA
e-mail: pzhang1@stevens.edu; jon.wade@stevens.edu; rich.trurner@stevens.edu

D. Bodner
Georgia Institute of Technology, Atlanta, GA, USA
e-mail: doug.bodner@gatech.edu

D. Thomas
University of Alabama, Huntsville, AL, USA
e-mail: dale.thomas@uah.edu

A.M. Madni et al. (eds.), *Disciplinary Convergence in Systems Engineering Research*, https://doi.org/10.1007/978-3-319-62217-0_80

challenging for educators to meet the growing educational demands for a workforce able to solve complex systems engineering problems [1–3].

Traditional techniques to assess competencies of systems engineering involve reviewing industry experiences together with written recommendations, which is very time consuming and is limited in accuracy. As systems engineering is an art as well as a science, capabilities are determined not only by knowledge but by skills and competencies [4, 5]. Assessing competencies of a candidate based solely on written statements and interviews is comparable to requiring drivers to only take the written test without road tests; information about the candidates' real-world performance will be lacking in this assessment. A new set of assessment techniques together with a comprehensive assessment model is needed to help fill the workforce gap by providing efficient and accurate assessment of systems engineering competencies both for existing systems engineers and those who are new to the field. Furthermore, an assessment method needs to be developed for the academic environment to assess systems engineering learning to provide feedback on instructional efficacy.

80.2 Systems Engineering Experience Accelerator

80.2.1 Introduction

The Systems Engineering Experience Accelerator (SEEA) project created a new approach to developing the systems engineering workforce, which augments traditional, in-class education methods with educational technologies aimed at accelerating skills and experience with immersive simulated learning situations that engage learners with problems to be solved. Although educational technology is used in a variety of domains to support learning, the SEEA is one of the few such technologies that support development of the systems engineering workforce.

The SEEA was developed to support a single-person role-playing experience in a digital environment, as well as a specific learning exercise in which a learner plays the role of a lead systems engineer for a Department of Defense (DoD) program developing a new unmanned aerial system. This exercise is based on the notion of experiential learning, and thus will be referenced as an experiential learning module. The learner engages with the experience (i.e., simulated world), makes decisions to solve problems, sees the results of those decisions, abstracts lessons learned from what was successful and what was unsuccessful, and then repeats the process in a series of cycles, simulating the evolution of the program over time.

The SEEA technology provides a graphical user interface allowing the learner to see the program status, interact with nonplayer characters to gain additional program information, and make technical decisions to correct problems. It also provides the capability to simulate the future behavior of the program, based on these learner decisions, so that outcomes can be shown to the learner. This cycle of decision and simulation-into-the-future supports the Kolb cycle of experiential learning [6]; the Experience Accelerator uses multiple such cycles operating through the lifecycle of the program. In particular, this approach allows communication of the effect of upstream decisions on downstream outcomes in the system lifecycle. The SEEA can support a wide variety of systems domains and areas of expertise through changes to the experience. Recently, additional multiplayer technology has been developed to allow live player support for team-based learning, as well as for a mentor or instructor to provide advice and feedback. The following are the problem statements and goals for this project.

Problem Statement Traditional systems engineering (SE) education is not adequate to meet the emerging challenges posed by ever-increasing systems and societal demands, the workforce called upon to meet them, and the timeframe in which these challenges need to be addressed.

Project Goal Transform the education of SE by creating a new paradigm capable of accelerating the time to mature a systems engineer while providing the skills necessary to address emerging system's challenges.

80.2.2 SEEA for Learning and Assessment

Learning assessment is a critical component of accelerated learning [7]. It is crucial to understand the learning results and the efficacy of different kind of learning experiences. This is imperative both in assessing the capabilities of the learner and in improving the efficacy and the capabilities of the learning experience. While assessment capabilities are critically important, nothing was found in the literature that was directly applicable to automated assessment of systems engineering skills in the SEEA. Therefore, a new experimental design grounded in the literature will need to be devised, along with a set of tools to facilitate its application.

While the Experience Accelerator (EA) has a broader goal of accelerating the learning of critical SE competencies through an experience-based system, systems thinking skills are a key component of the targeted learning outcomes. Systems thinking is at the core of the targeted EA SE competencies and therefore one of the primary competencies to be assessed in order to evaluate the effectiveness of the EA.

Systems thinking seeks to improve decision making and complex problem solving through deep systemic understanding. Typically, in order to assess learning gains in these areas, three approaches are utilized: measuring performance resulting from decisions (such as a game or simulation score), reviewing decisions and actions that were taken, or measuring learner understanding (the rules and mental operations that lead to decision making) [8, 9]. Measuring learner understanding seeks to verify that improved decision making arises from understanding the system and not simply from trial and error [10]. All of these approaches are valid and can result in worthwhile evaluation. As systems thinking skills are applied in order to understand and solve complex problems, educational research on the assessment of problem-solving skills can be helpful in designing an effective evaluation.

In order to solve an ill-structured problem, students must be able to deconstruct the problem into its constituent parts (e.g., stakeholders, relationships among them, impacts of the problem on them), define the problem in their own words, determine resources to help them understand the problem, determine and pursue learning issues, and develop and test a solution. Research on the evaluation of problem-solving skills tells us that in order to evaluate problem-solving ability, we must assess students' ability to do each of these steps. The EA seeks to accelerate the learning of novice SEs and advance them more quickly to expert SE performance. Experts use heuristics to skip steps; novices typically are not capable of doing this.

A meta-analysis of problem-solving assessment literature found that 18 of 23 studies deemed of high-quality use cases or simulations as assessment methods [11, 12]. With the EA simulation, we have the means to measure learner's performance within the experience. Learners make decisions within the EA, the simulation determines the results of those decisions, and we are provided with outcomes that we can utilize in order to assess the effectiveness of learners' decisions.

In order to assess learners' levels of understanding and to determine if the EA improves learning, a more thorough picture of the thinking behind learners' choices is needed. Therefore, to assess learners' understanding, it is important to elicit their views of the system, the problems they faced, and the thinking behind their decisions to solve these problems. Emerging literature in systems dynamics increasingly has instead been seeking to assess learners' understanding or mental models.

Therefore, learner performance assessment can be performed through analyzing the captured actions and decisions taken by the learner. EA captures learner approaches to decision making (through verbal protocols), and by using expert choices and protocols as a baseline for "good" decision making, one can assess learner understanding.

The evaluation plan therefore focused on:

- Benchmarking with an objective "score" which is also useful in motivating students
- Comparing subject matter expert (SME) EA actions and results to novice SE actions and results
- Comparing SME written (or transcribed verbal) descriptions of their decision-making process during the EA to novice SE written (or transcribed verbal) descriptions of their decision-making process during the EA in experience 1 and experience 2
- Tracking learning with changes in 1–3 above through a learner's multiple iterations through the experience

To support this plan, the EA has been instrumented to record information as a learning laboratory. Research will be done to determine the requisite data that needs to be recorded and the EA will be updated accordingly. Prior to completing this research, the following data has been selected and will be collected from the EA:

- Participant identification
  - Learner's name and demographic information
  - Team name and other members
  - Instruction name and roles played in experience
- Experience session information
  - Experience name and version
  - Date of experience start and end
  - Login dates and duration of each session
  - Phases/cycles covered in each login session
  - Elapsed time and number of session per phase/cycle
  - Links to past experience information
- Learner experience inputs and actions
  - Self-assessment
  - Initial recommendation input
  - All subsequent recommendation inputs
  - Workflow sequence with each action recorded with a timestamp
  - Who is called and which questions are asked, in which order
- Instructor input
  - Feedback provided to learners (dialog, email, etc.)
  - Recommendations accepted/rejected
  - Instructor's observations
- Simulation output
  - Last phase/cycle completed
  - Results of schedule, cost, range, and quality
  - Final status charts
  - Final score
- Reflection
  - Reflection feedback provided to the learner
  - Learner's reflection input

Next, a set of analysis tools are being developed to analyze this information. Test cases are being created to provide benchmarks to baseline this analysis. Finally, a demonstrable set of learning experiences will be recorded and analyzed to provide feedback on the capabilities of the system.

80.3 The Learning Experience

80.3.1 Learning with the SEEA

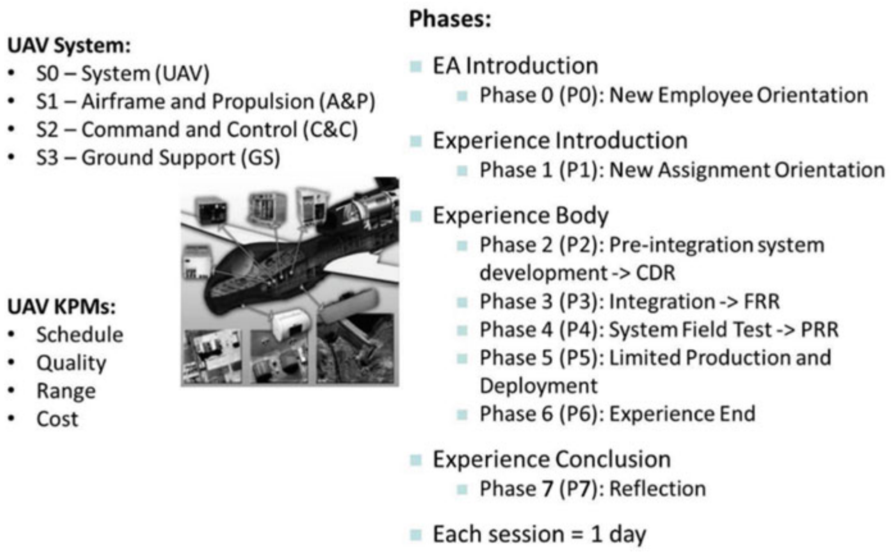
The Systems Engineering Experience Accelerator provides the capability to simulate the program into the future, based on these learner decisions, so that outcomes can be presented to the learner. This cycle-based decision-making process and simulation-into-the-future supports the Kolb cycle of experiential learning [6]; the Experience Accelerator uses multiple such cycles operating through the life cycle of the program. Specifically, this approach allows illustration of the effect of upstream decisions on downstream outcomes in the system life cycle.

Applied in an academic setting, the SEEA concept provides the possibility for a much broader scope of learning environments than a capstone project or industry internship [13]. These more traditional approaches provide a beneficial learning experience and support integrating the various components of the SE body of knowledge, but are limited by time and domain. The capstone is usually a single project and at most a year in length. If it covers the full life cycle, then it must be a simple project and most likely represents only one domain. An internship is even more limited, given that few companies would assign a student to a significant role or provide much variation of role or domain. The SEEA envisions the ability to provide learning experiences that involve significant decision making at various levels of authority and drawn from many different domains. Neither a capstone nor an internship could likely present the same range of specific challenges and "aha" moments that the SEEA can provide. Whether the SEEA experience is as effective as a truly in vivo experience is part of the research underway, with results from academic and industrial pilots of the SEEA as the primary means of validating effectiveness.

80.3.2 The Current Learning Experience

The current SEEA learning experience was designed in a defense acquisition program context [12] where an unmanned aerial vehicle (UAV) acquisition program is underway. The learning experience utilizes the following scenario.

The XZ-5 is a sophisticated UAV system being developed for all services for reconnaissance, surveillance, and targeting missions. In this experience, the lead learner assumes the role of the lead systems engineer just after the preliminary design review, replacing the previous lead program systems engineer. The XZ-5 project completed a technical development phase and preliminary design review (PDR) in the second fiscal year (FY-2) and entered a cost-plus-fixed-fee (CPFF) contract for engineering and manufacturing development (EMD) phase after a favorable Milestone B (MSB) decision. The contract budget base is $200 M with $195 M initially allocated to the performance measurement baseline (PMB). The



UAV System:
- S0 – System (UAV)
- S1 – Airframe and Propulsion (A&P)
- S2 – Command and Control (C&C)
- S3 – Ground Support (GS)

UAV KPMs:
- Schedule
- Quality
- Range
- Cost

Phases:
- EA Introduction
  - Phase 0 (P0): New Employee Orientation
- Experience Introduction
  - Phase 1 (P1): New Assignment Orientation
- Experience Body
  - Phase 2 (P2): Pre-integration system development -> CDR
  - Phase 3 (P3): Integration -> FRR
  - Phase 4 (P4): System Field Test -> PRR
  - Phase 5 (P5): Limited Production and Deployment
  - Phase 6 (P6): Experience End
- Experience Conclusion
  - Phase 7 (P7): Reflection
- Each session = 1 day

Fig. 80.1 Context for the UAV experience [7]

program is supported by a prime contractor and three major subcontractors. The experience starts with the beginning of FY-3, just after the EMD contract is awarded. The learners' team has just checked on board to the XZ-5 government program office. The XZ-5 program manager is counting on the team to establish "ground truth" on the technical status and trajectory of the XZ-5 development and make recommendations to keep the program on track to enter critical design review (CDR) on time at the end of FY-4.

The current XZ-5 project under development consists of three major subsystems: The airframe and propulsion is primarily electromechanical, the command and control system is mainly software, and the ground support system is mainly human based. The key performance measures (KPMs) are schedule, quality, range, and cost. Each of the learner's sessions in the experience represents a single day in the program and is estimated to take approximately 1 h to complete, although the learner is free to log in and out any number of times during a session (Fig. 80.1).

80.3.3 Pilot Use of the Learning Experience in a Project Management Course

More than 30 junior and senior engineering undergraduates at the University of Alabama in Huntsville (UAH) used the SEEA during the 2016 spring semester as a team project. The students were enrolled in the Management Systems Analysis course, which focuses primarily on project management skills. Students were asked to participate in teams of five. Each student in a team plays a different role in the XZ-5 UAV experience. Those roles include lead systems engineer (LSE), airframe and propulsion system (APS) lead, command and control system (CCS) lead, ground stations launch and retrieval system lead (LGLRS), and integration lead (Prime). Each team was tasked with using the SEEA in the UAV scenario given as two homework assignments – one near the beginning of the semester, and one near the end of the semester to evaluate the students' skill advancement.

Phase 0 introduces the students to the SEEA and the XZ-5 program; phase 1 explains to the students their new assignment; phase 2 requires the students to analyze the current situation just after the completion of the preliminary design review (PDR) and make recommendations to keep the program on track, leading to the critical design review (CDR); phase 6 provides the results of the current simulation based on the performance of the students; and phase 7 gathers information and provides feedback to the students based on their actions taken during the experience and reflect on learning skills. Phases 3, 4, and 5, simulating integration, system test, and limited production and deployment, are currently being updated.

80.4 Results and Analysis

80.4.1 Pilot Results

After the pilot course was completed, the performance data of the teams were gathered and compared. Due to technical difficulties in the first run, only the results from the second run of the SEEA are used in this analysis. The performance measures include range, critical software defects, schedule, CDR artifact completion, and budget overrun. The SEEA combines these measures to determine if the CDR can be achieved successfully and determines the risk to proceed with the UAV program. During the pilot, each of the seven teams made different decisions, resulting in a range of performances and different program results. Among the seven teams participating, five teams were able to complete the whole project cycle and reach phase 7 to receive performance feedback from the SEEA. Teams 1, 5, and 6 all finished with a low risk of proceeding based on CDR results; team 3 finished with medium risk and team 2 finished with high risk. Teams 4 and 7 did not complete the simulation.

The data gathered during the pilot application can be analyzed to provide insights on students' decision making, their capability to discover issues in the system, their ability to prioritize resources and the outcomes of their decisions. As mentioned in Sect. 80.2.2, many different types of data were gathered by the SEEA system. Participant identification and experience session information are used to identify specific user and their use of the system. Learner experience inputs and actions are valuable data to track the learner's actions and behaviors during the experience, which will provide insights into the learner's decision-making process.

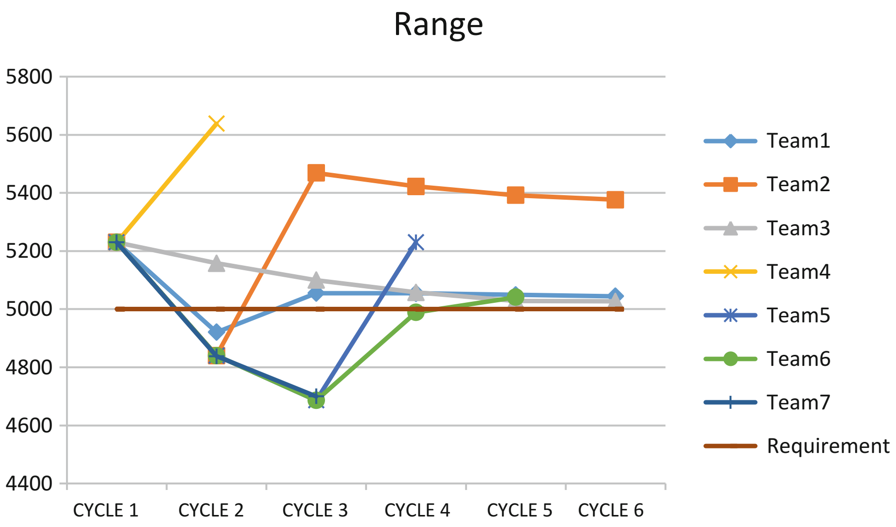


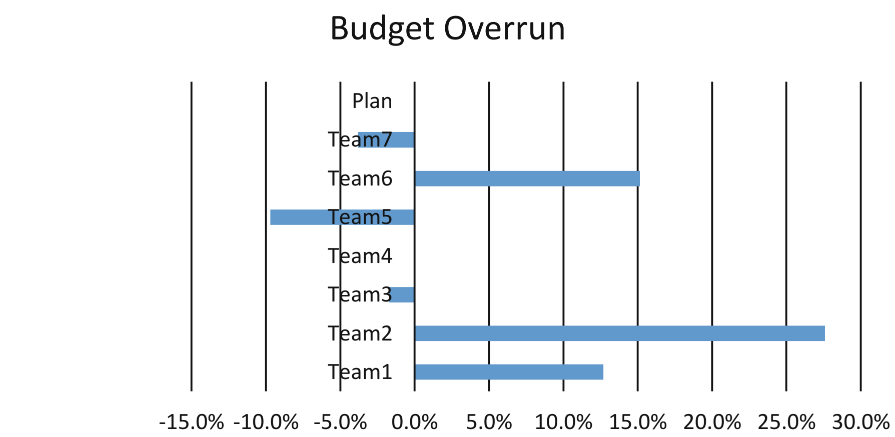
Fig. 80.2 XZ-5 UAV range performance

Simulation output data was used to determine the general performance for the learner; it also demonstrates the outcomes of learner decisions. Instructor input and reflections can be used to evaluate the efficacy of the learning and to improve the learning experience.

The performance of the teams is shown in Fig. 80.2. Range of the UAV is affected by weight, drag coefficient, and thrust-specific fuel consumption (TSFC). Team 2 performed very well with range, whereas teams 1, 3, 5, and 6 achieved the requirement. In the beginning of the experience, there were early signs of a range problem caused by weight issues, and most of the teams identified this issue by reallocating the weight balance and adding more workforce to the airframe and propulsion team.

Budget is an important measure to the success of the UAV program. Teams need to control the budget to be successful in the experience. While team 2 performed well in range, the recommendations they made caused a significant budget overrun. All the successful teams managed the budget and had a budget overrun of less than 15%. Figure 80.3 shows the overall budget overrun performance for the pilot application.

The XZ-5 UAV program has an original plan of 27 months between PDR and CDR. Any significant delay will potentially undermine the success of the program. It is recommended by the experts that the schedule shall not be delayed over 20 months while the delay within 10% of the period is considered good. Teams that manage the schedule well are likely to pass CDR proceed with low risk. Teams 3, 5, 6, and 7 managed the schedule well. Team 4 recommended advancing the CDR time by 5 months, which resulted in incomplete work. Teams 1 and 2 performed within acceptable range (Fig. 80.4).



	Team1	Team2	Team3	Team4	Team5	Team6	Team7	Plan
■ Budget Overrun	12.702%	27.589%	-1.735%	-0.026%	-9.713%	15.141%	-3.808%	0.000%

Fig. 80.3 XZ-5 UAV budget overrun performance

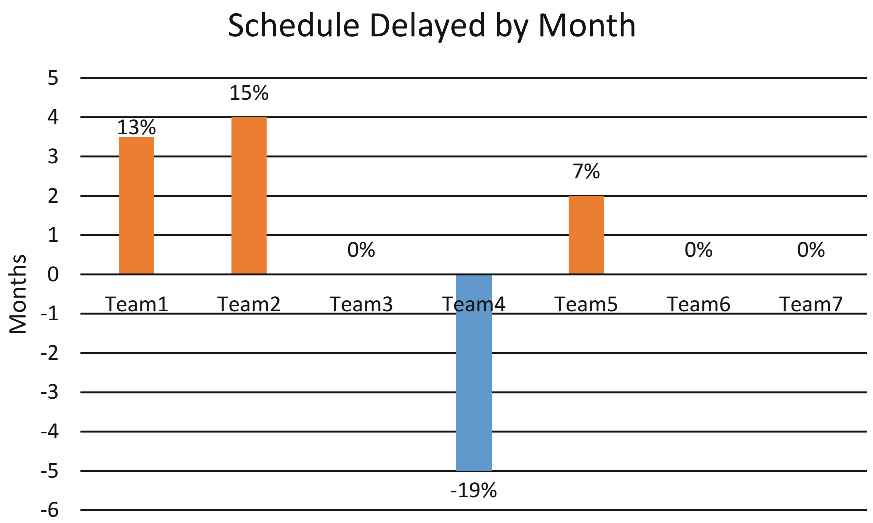


Fig. 80.4 XZ-5 UAV schedule performance

Another performance measure was software critical defects; these indicators are affected by the mix of senior-junior staff and the number of software reviews. It is recommended to have less than eight critical defects to pass CDR proceed with low risk. Teams 1, 5, 6, and 7 kept the critical defects quite low, while teams 2 and 3 kept them controlled (Fig. 80.5).

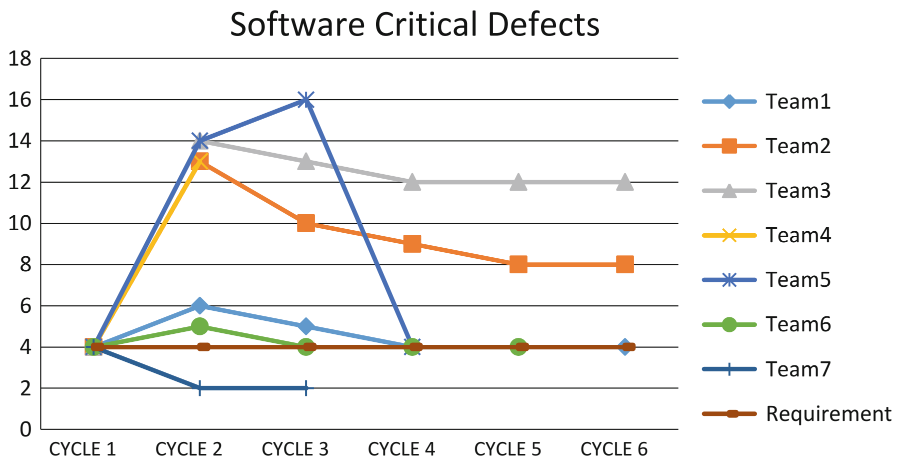


Fig. 80.5 XZ-5 UAV software critical defects performance

80.4.2 Pilot Analysis

As mentioned in Sect. 80.4.1, seven teams performed quite differently throughout the experience. Based on the data gathered from the SEEA, their downstream performance reflected their decision-making capabilities at crucial points in the project. The simulation challenged students to take on a project that has existing issues from the previous development phase and thus requires them to make changes to the system and project quickly and accurately. The teams that reacted more quickly in the right direction performed generally better than the teams who simply observed the situation without making the necessary changes. Table 80.1 shows the performance of the different teams along with their presentation results, decisions, and actions throughout the experience. Team scores were calculated using a weighted system based on the learners' performance on schedule, range, budget, software critical defects, and CDR readiness. The scores were normalized such that a score of zero would be equivalent to making no changes in the program and 100 was the best score that experts were able to achieve.

80.5 Summary and Future Works

This paper discussed the use of Systems Engineering Experience Accelerator (SEEA) in the domain of systems engineering (SE) education and learning assessment. During the pilot application of the technology, data was gathered from seven teams of students who participated in the UAV learning experience. Data gathered from the pilot application provided insights into the students' decision making and their understanding of systems engineering and project management. The technical

Table 80.1 Students' input and reflection

Teams	Simulation result and score	Presentation results	Decisions and actions throughout the experience
Team 1	Finished the experience, entered CDR with low risks. Program completed successfully Score 83	Decisions that would be changed in hindsight: Command and control weight would have been decreased more significantly More junior staff would have been hired and less senior staff to avoid costs Overall, the project was overrun by 13% at the end of phase 2, so more questions would have been asked to stakeholders to make better decisions	Increased the CCS weight allocation Increased senior staff and decreased junior staff Increased the drag coefficient target
Team 2	Finished the experience, entered CDR with high risks. Program canceled Score 58	N/A	Increased the CCS weight allocation and hired more junior staff More junior staff and increased the drag coefficient target significantly Decreased CCS weight allocation and hired even more junior staff Changed senior/junior staff mix
Team 3	Finished the experience, entered CDR with medium risks. Program terminated Score 77	Entry criteria for CDR were not achieved due to personnel disbursement error. After hiring and training new personnel, it was decided to move forward in the hopes of achieving at least 80% effectiveness In hindsight, the team would ensure the correct amount of personnel per department is hired and trained efficiently and effectively to meet guidelines and quality metrics for the success of the program	Decreased CCS weight allocation and increased both senior and junior staff Decreased CCS weight allocation. Further increased senior and junior staff. Increased drag coefficient target Further increased senior and junior staff
Team 4	Didn't finish the experience Score N/A	Most likely would not have been ready for the CDR because of the issues with scheduling and project progress, but there seems to be improvement compared to our previous run We were more willing to	Increased senior staff in APS and CCS, change weight allocations Added more senior staff and less junior staff

(continued)

Table 80.1 (continued)

Teams	Simulation result and score	Presentation results	Decisions and actions throughout the experience
		make changes this time, which seemed to improve the project overall	
Team 5	Finished the experience. Entered CDR with medium risks. Program terminated Score 44	Our CDR was delayed by 2 months because the range wasn't where we wanted it to be After delay, CDR criteria were achieved and we proceeded to the next phase CDR completed and mission accomplished	Increased senior and junior staff More senior staff and less junior staff. Increased drag coefficient target Increased senior and junior staff
Team 6	Finished the experience. Entered CDR with low risks. Program completed successfully Score 86	Adding quality engineers was very successful in our simulation	Increased senior and junior staff Decreased senior staff slightly Increased weight allocation for CCS. Increased target of drag coefficient Reduced junior staff number
Team 7	Didn't finish the experience Score N/A	Would do differently: Add more staff to APS at the beginning to reduce the drag Not hire as much staff for the CCS Try to find different ways to reduce the drag coefficient Try to find different ways to increase the range	Reduced the total weight allocation of APS, increased both senior and junior staff for CCS, increased the GS junior staff, increased APS Added more senior and junior staff. Increased software review frequency

difficulties encountered in the first run of this pilot have been resolved, so for future pilot applications, multiple runs of the SEEA will be performed and compared for performance analysis. While there were technical issues during the pilot application, SEEA was unanimously praised by the students in that it provided an opportunity to practice the skills that were illustrated in the classroom.

The future works for this research include gathering data through pilot application with a number of systems engineering experts; using data gathered from expert pilot use of SEEA to calibrate the experience and scoring mechanism; comparing students' behavior data and decision-making process with experts'; and pilot applications with two separate runs of the SEEA before and after the learning and using the data gathered to assess the efficacy of the learning.

The SEEA will be utilized for another pilot application in a graduate Introduction to Systems Engineering course at UAH in the Fall 2016 semester and will be utilized again in the Management Systems Analysis course in the Spring 2017 semester.

Acknowledgments This material is based upon work supported, in whole or in part, by the US Department of Defense through the Systems Engineering Research Center (SERC) under Contract H98230-08-D-0171. SERC is a federally funded University Affiliated Research Center managed by Stevens Institute of Technology. The support and guidance by the Defense Acquisition University, Dave Pearson, John Snoderly, and Scott Lucero were critical to this program. The authors thank all members of the Experience Accelerator team, as well as the subject matter experts who have provided guidance for authenticity of the Experience Accelerator. The authors particularly wish to thank the students within the University of Alabama Huntsville course.

References

1. Bagg TC (2003) Systems engineering education development (SEED) case study
2. Davidz HL, Nightingale DJ (2008) Enabling systems thinking to accelerate the development of senior systems engineers. Syst Eng 11(1):1–14
3. Squires Aet al (2011) Building a competency taxonomy to guide experience acceleration of lead program systems engineers, DTIC Document
4. Hutchison N et al (2014) 7.2.3 early findings from interviewing systems engineers who support the U.S. Department of Defense. INCOSE Int Symp 24(1):643–657
5. Pyster A et al (2014) Atlas: the theory of effective systems engineers, Version 0.25
6. Kolb DA (1984) Experiential learning: experience as the source of learning and development. Pearson Education, Upper Saddle River
7. Wade J et al (2015) Developing the systems engineering experience accelerator (SEEA) prototype and roadmap
8. Karakul M, Qudrat-Ullah H (2008) How to improve dynamic decision making? Practice and promise. In: Complex decision making. Springer, New York/Berlin, pp 3–24
9. Rouwette EA, Größler A, Vennix JA (2004) Exploring influencing factors on rationality: a literature review of dynamic decision-making studies in system dynamics. Syst Res Behav Sci 21(4):351–370
10. Kopainsky B, Pirnay-Dummer P, Alessi SM (2012) Automated assessment of learners' understanding in complex dynamic systems. Syst Dyn Rev 28(2):131–156
11. Belland BR, French BF, Ertmer PA (2009) Validity and problem-based learning research: a review of instruments used to assess intended learning outcomes. Interdis J Probl-Based Learn 3(1):59
12. Bodner D et al (2012) Simulation-based decision support for systems engineering experience acceleration. In:Systems conference (SysCon), IEEE International.IEEE
13. ZhangP et al (2016) The experience accelerator: tools for development and learning assessment. ASEE Conferences, New Orleans

Chapter 81
Future Systems Engineering Research Directions

Jon Wade, Rick Adcock, Tom McDermot, and Larry Strawser

Abstract This paper describes a program organized by the INCOSE Academic Council to determine future directions in systems engineering (SE) research. This program, consisting of three collaborative workshops, uses a framework coupling societal needs to systems challenges and then to gaps in the capabilities of SE, which inform the direction of future SE research. The results of the first workshop are presented including a description of the Grand Challenges in five selected areas, namely, societal needs, problem definitions, desired results, obstacles, and related research questions. This paper concludes with a summary and description of the future work for this program.

Keywords Systems engineering • Systems engineering research • Grand Challenges

81.1 Introduction

We choose to go to the moon. We choose to go to the moon in this decade and do the other things, not because they are easy, but because they are hard, because that goal will serve to organize and measure the best of our energies and skills, because that challenge is one that we are willing to accept, one we are unwilling to postpone, and one which we intend to win, and the others, too. John F. Kennedy [1]

J. Wade (✉)
Stevens Institute of Technology, Hoboken, NJ, USA
e-mail: jon.wade@stevens.edu

R. Adcock
Cranfield University, Cranfield, UK
e-mail: r.d.adcock@cranfield.ac.uk

T. McDermot
Georgia Tech, Atlanta, GA, USA
e-mail: tom.mcdermott@gtri.gatech.edu

L. Strawser
Johns HopkinsUniversity, Baltimore, MD, USA
e-mail: lstraws1@jhu.edu

A.M. Madni et al. (eds.), *Disciplinary Convergence in Systems Engineering Research*, https://doi.org/10.1007/978-3-319-62217-0_81

The use of Grand Challenges to focus technical efforts is not new. One could argue that the construction of the Grand Pyramid of Giza over a period of 20 years in the time of 2500 BCE represented such a systems engineering Grand Challenge, as did the other Seven Wonders of the World. Much more recently, in 1900, David Hilbert, a famous mathematician, formulated the now-canonical "23 Problems of Hilbert" that served to focus the efforts of mathematics in the twentieth century [2]. John F. Kennedy provided the same inspiration and focus with the race to the moon that provided a focal point for so many technical fields in the 1960s, establishing the basis for Moore's law [3] in electronics whose exponential advances have shaped our present world. Other Grand Challenges, such as in global health, have been funded by the Bill and Melinda Gates Foundation [4] with the objective of focusing science and technology on accelerated developments against diseases that affect the developing world. The National Academy of Engineering has also created a list of 14 Grand Challenges in a diverse set of areas in critical technical areas [5]. Other challenges include those from XPRIZE [6], Longitude Prize [7], and the Grand Challenges of Social Work [8].More specifically in the area of systems engineering, Kalawsky proposed research in the five defined areas in systems engineering[9].

This project was established by the Academic Council of the International Council on Systems Engineering (INCOSE). The Academic Council is a branch of the INCOSE Corporate Advisory Board facilitating discussion and exploration of issues relevant to academia and setting a path for achievement supported by strategic collaborations.

The project utilizes the approach described in the publication, "A World in Motion: Systems Engineering Vision 2025" [10], which uses a framework coupling societal needs to systems challenges and then to gaps in the capabilities of systems engineering. Primary references for this work are in the NAE Grand Challenges [5] and the World Economic Forum's "The Global Risks Report" [11]. The objectives of this project are to:

- Build communities among academia, industry, and government from numerous domains, expanding beyond defense/aerospace, dedicated to tackling the major global systems challenges confronting humankind for societal good
- Excite, inspire, and guide SE research in these communities
- Achieve consensus among these communities to establish the priorities of SE research
- Provide the means by which to create synergy in these SE research efforts such that progress can be measured against these objectives

81.2 Project Structure

As described in [10], this project will (1) couple societal needs to systems Grand Challenges, (2) determine the gaps in systems engineering and science necessary to address these challenges, and from this analysis, and (3) determine critical areas of systems research, resulting in a proposal for a roadmap of systems research. This project will be conducted as a series of workshops, with each workshop focused on one of these critical areas, building a connection to the next workshop. The workshops will be moderate in size with 35–50 attendees representing academia, government, and industry from a range of domains beyond the traditional domain of DoD/aerospace. Tentatively, the project will consist of three workshops, followed by a symposium, perhaps coincidental with the INCOSE International Symposium (IS).The tentative foci of the three workshops are:

- Workshop I: Review the program framework (based on the INCOSE SE Vision 2025) and define a set of high-level societal needs and Grand Challenges to provide focus for the SE research. The workshop was conducted at Johns Hopkins University Advanced Physics Laboratory on October 12–13, 2016. The results were presented at the 2017 INCOSE International Workshop.
- Workshop II: Determine the SE capabilities and gaps necessary to address the Grand Challenges defined in Workshop I. Scheduled for March 22, 2017, preceding the Conference on Systems Engineering Research (CSER) held at USC.
- Workshop III: Determine the areas of research necessary to address the SE capabilities and gaps defined in Workshop II. Tentatively scheduled for early summer 2017.
- Symposium: Communicate the results of the SE research and kick off community building efforts at INCOSE International Symposium July 17–20, 2017 in Adelaide, Australia.

This paper focuses on the results of Workshop I which is on the identification of societal needs and the specification of Grand Challenges necessary to address them.

81.3 Workshop I

Approximately 35 attendees who are known to be thought leaders in systems and engineering were invited from industry, government, and academia. The workshop was comprised of four breakout sections, each composed of 6–8 attendees to address a separate domain of interest. Participants in this workshop were encouraged to rank in order their interest in the societal needs from the SE Vision 2025 as noted below. The responses were scored with 7 points for the first choice, 6 points for the second choice, and so on. The four bolded societal needs areas noted below were selected. The nonselected areas may be revisited in future academic research forum discussions. Based on this, participants were assigned to groups and were notified prior to attending the workshop:

- *Food and clean water – selected*
- Healthy (physical) environment
- *Access to healthcare – selected*
- *Access to education – selected*
- Access to information and communication
- Transportation and mobility
- *Security and safety (physical and cyber) – selected*
- Others (attendee's choice)

The workshop was initiated by keynote presentations by Kristen Baldwin, Dennis Buede, and Rick Adcock, describing the Grand Challenges of national security, industry, and academia, respectively. Then each breakout group spent the afternoon of Day 1 describing their area of focus in their selected societal needs domains. At the end of Day 1, each group presented their results to the rest of the participants for comments and feedback. On Day 2, each breakout group refined their area of focus and formulated a description of a systems Grand Challenge necessary to address the described societal needs.

The following are the desired characteristics of the societal needs Grand Challenges [12]:

1. Represent complex and extremely difficult questions that are solvable (potentially within 10–20 years).
2. Improve quality of life through positive educational, social, and economic outcome potentially affecting millions of people.
3. Involve multiple research projects across many subdisciplines to be satisfactorily addressed.
4. Require measurable outcomes so that progress and completion can be identified.
5. Compel popular support by encouraging the public to relate to, understand, and appreciate the outcomes of the effort.

To this list, we add the requirement that the Grand Challenges are systemic in nature and cannot be solved by technology alone. The results of the workshop are described below.

81.4 Workshop I Results

81.4.1 Clean Water

81.4.1.1 Grand Challenge

After deciding to focus entirely on clean water, this group determined their Grand Challenge to be: *Ensure that economical access to locally sourced, clean water* *necessary for survival and limited prosperity (as a minimum requirement) is available to everyone*. A number of actions need to be taken to achieve this objective addressing water quantity, quality, and location. Innovation is necessary to create new methods to develop water availability. Appropriate weather responses need to be made to ensure that the available freshwater is preserved for use. Actions need to be taken to prevent the decline of water quality. Water supplies need to be made secure, ensuring that hostile entities are not able to have a negative effect. It will be critical to work cooperatively with commercial entities in these areas.

81.4.1.2 Problem Definition

Four billion people have an inadequate water supply at least part of the year [13]. Several "water problem types" need to be considered to explore this, including scarce water, plentiful but contaminated water, or seasonal water presence. Contamination might be created from both natural- and human-caused events, the latter from unintended and intended actions. Due to the wide variation in freshwater supply challenges, there is no single solution that can address the Grand Challenge in all environments. Hence, the solution sets will need to be context aware, meeting the specific sociotechnical conditions in each locale.

81.4.1.3 Desired Results

The challenge set by this group is focused on a minimum set of outcomes, which take the lack of clean water out of the critical path for other economic development concerns. Non-contaminated water is available to everyone from local sources, generated in a way that is environmentally friendly. The solution sets will be scaled to target population requirements and cleanliness standards. The solution sets will avoid requiring capital-intensive new infrastructure or reliance on existing transportation systems for everyone. The solution sets must be feasible, cost-effective, and adapted to the region. Note that other clean water-related issues such as access to external supplies, decisions as to where to build new developments, etc. could also be considered by more developed nations but are outside of the scope of the challenge set by this workshop.

81.4.1.4 Obstacles

Geopolitical, cultural, and other social factors may inhibit required local collaboration and cooperation to implement solutions. Privacy issues may reduce the ability to collect necessary data needed to implement solution sets. Political turmoil, social unrest, corruption, and the lack of a stable governing body can all greatly reduce the ability to positively affect change. Extreme poverty and the lack of a viable economy may retard the deployment of viable systems. Cultural reluctance to embrace externally generated solutions may prevent adoption. People may be residing in areas that have dwindling or nonexistent water supplies due to climate change. Finally, adequate models do not exist to adequately support the interdependencies of this complex systems problem.

81.4.1.5 Research Questions

The group identified a number of research questions including:

- Can a systems of systems model to simulate contamination of water sources be developed to support option generation and decision-making, including the following:
  - Man-made or natural, intentional or unintentional, contamination of global water sources
  - Continental, regional, state, territorial, or tribal scale
- Can the solution sets incorporate policies and regulatory guidelines and include measurable outcomes?
- Can tabletop exercises be developed that show factors such as response time, recovery achieved (percent as partial, total, etc.), resulting cost, etc.?

81.4.2 Healthcare

81.4.2.1 Grand Challenge

The group identified a number of specific challenges with the provision of healthcare and more generally with the difficulty of considering this type of challenge at a national or international level. Given this, the group identified the following generic challenge: *How can we frame the issues around healthcare as a system in such a way that they can be considered at a national or international level?* The discussions of this challenge focused on understanding an enterprise such as healthcare, which is judged not only as a complex human and technology system but also within the context of the society within which the assessment is being made.

81.4.2.2 Problem Definition

The following working definition of healthcare was created:

The ability to *restore* or *maintain* the *health* of the mind and body of citizens of a society – this includes *well-being*, *treatment*, and *social care* – applied to individuals, groups, society, etc.

All of the terms in bold above would need to be further defined, and this definition depends on the social context in which healthcare sits. A number of specific healthcare challenges were discussed, including:

1. How to ensure a basic level of health within the community of a developing country, sufficient to allow other aspects of social and economic growth.
2. How to provide adequate social care to members of the society, in particular, one in which nutrition, prevention, and treatment have led to an increasing number of people in need of such care to maintain quality of life.
3. How to balance a basic level of health maintenance across a society with the ability to offer a full range of available healthcare for those more able to bear some or all of the cost.
4. How to ensure the correct balance between self-care, e.g., good diet, regular exercise, health monitoring, etc., and regulated social care.

While any of these could form Grand Challenge themes, the workshop group decided to focus on the generic issue of understanding such questions in a broad social context.

The workshop group used the term *socioeconomic enterprise* as a working label of something of the scope and complexity of healthcare; see "Obstacles" section below. Such an enterprise must have a scope which covers all aspects of its outcomes, e.g., medical care and individual responsibility. It must be able to balance finite resources and make trade-offs acceptable within the social context in which it sits. It must deal with the human and technical complexity of integrating and operating multiple service systems, e.g., combining well-established clinical practices with a wide range of technologies. It also needs to remain viable when faced with all of the environmental factors and possible threats which exist now and in the future in the defined context.

81.4.2.3 Desired Results

The desired results would be the resolution of a number of social, technical, and economic outcomes. Society begins to think of healthcare as an overall system, considering the complexity of all the issues raised and making balanced judgments. This provides the basis to integrate all the disparate pieces of our healthcare enterprise, ensuring all component parts can operate effectively but gain additional value from considering it as a whole. We obtain and make clear an accessible appropriate information about all aspects of healthcare: individual status, best evidence, options, trade-offs, and associated costs. Through this, we drive investment and research activity better to improve the quality of health and decrease the cost over time.

81.4.2.4 Obstacles

The group spent some time considering the systemic nature of a challenge such as healthcare. Healthcare delivery is provided by a number of related "sociotechnical" systems (systems combining people and technology toward a socially or commercially agreed goal or service). Many of these could be described as "cyber-physical service systems" (physical systems in which software and communications technology play a key role in how the system works, often allowing for open flexible elements to be brought together by the end user at the point of operation to provide services) and "systems of systems" (in which, alongside the issues of technical and human integration, we must deal with component systems with independent missions, owners, and change cycles).The above simplified definitions are based on the SE Body of Knowledge [14].

In general, it was thought that such systems not only face significant issues in overcoming technical and human integration issues, but also that it is difficult for the society in which they sit to agree on the relative value of the services they provide and to correctly assign the necessary resources to achieve those values.

The challenges of integrating the elements of "cyber-physical systems" and "systems of systems" are already documented. In healthcare, this will be further complicated by the need to work within established clinical and public health practices. Many of these practices are based upon technological constraints which may no longer exist but which are difficult to untangle from current ways of working.

This is an example of a wicked problem [15] and as such all possible solutions have both positive and negative outcomes. In the face of this complexity, it is also very difficult to have a "grown-up conversation" about them and to avoid political or commercial interest groups setting the agenda for such discussions.

81.4.2.5 Research Questions

From the discussions above, the group identified the following initial research questions:

1. How to model a socioeconomic enterprise such as healthcare in a way that fully explores missions, context, environment, interacting elements, etc., and is accessible to a wider community to facilitate a balanced conversation.
2. How to use this understanding to set goals, incentives, and priorities and use these to drive and measure change.
3. What are the ways of describing such an enterprise using system architecture models and using this to significantly improve integration both within and across the component systems.
4. How to take particular views of the enterprise, for example, information sharing and use, performance, people and skills, resilience and security, etc., and use these to generate options for change.

5. How to identify ways to assess the architectures and options in 3 and 4 above against the measures described in 2 and in the context of broader understanding established by 1.

All of these use aspects of existing SE but apply them to a scale, complexity, community, etc. not currently considered.

81.4.3 Education

81.4.3.1 Grand Challenge

The group identified a single, broad Grand Challenge in education which is to *reform education systems to address gaps in systems skills in individuals.* In this case, systems skills include a very detailed set of knowledge, skills, and abilities (KSA) including technical, managerial, professional, and crossdisciplinary per the INCOSE competency framework categories currently being developed. This Grand Challenge is a superset of the three major SE Vision 2025 goals of ensuring that all systems decision-makers are systems thinkers, that all engineers have systems engineering skills, and that all systems engineers are broad-based, technical leaders. Achieving this Grand Challenge will require the identification of an appropriate set of systems skills supporting societal needs, a determination of the skill gaps in existing workforce and graduates and the systematic reform of the education system to address those gaps.

81.4.3.2 Problem Definition

There are numerous critical issues in education systems throughout the world, with specific problems depending on the specific context. In many cases, current education systems were developed to support the needs of early industrialization and are far short of providing students with the KSAs that are required in today's information-based, networked societies. In addition, education systems often do not provide their students with the capability of working together effectively with an appreciation of diversity of gender, backgrounds, environments, and technology. While our societies at both the local and global levels are becoming more tightly interconnected as systems of systems, current education systems generally are not systems aware and do not instill systems literacy in their students. Finally, education systems tend to be quite resistant to change and are often constrained by entrenched constituencies and bureaucracy.

81.4.3.3 Desired Results

The desired result is that students increasingly (up through graduation and on the job) are capable of providing immediate and long-lasting benefit to society, can work collaboratively with individuals of diverse backgrounds and environments, have a systems perspective, and have professional skills such as ethics, communications, leadership, and followership. To make this possible, everyone associated with education will have the necessary KSAs, tools, and techniques to make sense of society and how they can provide immediate and long-lasting value. These educational capabilities need to be supported with the ability to effectively assess the level of KSAs in students and citizens, and the means by which to make this available to a populace willing to use these results for their own betterments. Finally, society will have the ability to use systems methods to better understand and appreciate diverse value systems, thus facilitating collaborative work in our networked society.

81.4.3.4 Obstacles

Education systems are perhaps one of the most difficult systems to change due to the size and scale of often entrenched bureaucracies that support them, the long lag time between change initiation and effected change in the resultant workforce, the often conflicting objectives of the various stakeholders, and the difficulty in measuring educational results and effectiveness. Unfortunately, this is the case where attempts to measure educational results change the education process as evidenced by standardized testing in the United States. There are numerous obstacles including the fact that society does not have a common understanding of the concept of systems, does not think in a systems manner, and may not recognize the advantage of this approach. Industry and government may not be willing to share KSA information in a form that is readily useful, and even if there is a will, privacy issues may restrict the ability to collect data. Legacy education systems are often very complex, are distributed, and are resistant to change. Finally, major changes may be necessary as the current education system is inadequate in preparing graduates to provide immediate and long-term value to society.

81.4.3.5 Research Questions

There are numerous research questions that arise based on the aforementioned Grand Challenge objectives and the obstacles noted above. Questions range from the ability to determine KSAs that we need based on the global and local contexts, how they can be taught and evaluated, and how education systems can be changed and made adaptive and resilient. The following are examples of some specific research questions:

- Do systems skills result in better SEs, engineers, and citizens?
- Is it possible to define the systems skills and KSAs that cover all the roles that an SE, engineer, or citizen may take?
- How dependent is this on cultural and societal value systems?
- Is it possible to unambiguously determine the gap between education and desired KSAs? If so, is it possible to define a course of study/training/internship to address these?
- Is it possible to effectively and efficiently determine the means by which to evaluate someone's capabilities with respect to each KSA?
- Is it possible to assemble an education system with the materials, tools, staff, and organizations to support the desired actions and outcomes?
- Does a project team-based curriculum, based on multidisciplinary systems thinking, result in learning that is better than current learning as evaluated by standardized assessment?
- How can "internship" concepts and best practices be used effectively throughout the educational process (from cradle to grave)?
- How effectively can the educational process be studied and modeled as a system?
- How can lifelong educational and training systems be updated and made to be resilient and adaptive?

81.4.4 Security and Safety

81.4.4.1 Grand Challenges

The group identified two Grand Challenges, which were both derived from the National Academy of Engineering (NAE) Grand Challenges *Prevent Nuclear Terror* and *Secure Cyberspace*. These were generalized as *resilience to catastrophic events* and *systemic security*. For the first, the generalization of resilience to catastrophic events represents the need for systems engineering methods, processes, and tools that adequately capture both the system and its external context, in order to address systemic effects. Because such events are necessarily addressed by combined policy, economic, and technical strategies, future systems engineering tools must capture both technical and nontechnical solution sets. For the second, there is a recognition that the traditional methods of hierarchical decomposition of structure and associated perimeter defense strategies are no longer valid. Future systems engineering tools need to consider the heterarchical nature of cyber systems and their context.

81.4.4.2 Problem Definition

The advancement of technology and information access has lowered threat barriers to entry, and security challenges are diffusing across all domains (many of which have not been designed to be secure). There is a need to reevaluate how we design systems in response to real and potential threats. Here we define a threat in a general sense: anything that would disrupt, damage, or destroy the system or its stakeholders. Systems that were developed without any consideration of operational threats now are being disrupted, and systems are being used by threat actors in ways that were never intended uses of the system. Thus the consideration of system response to threats in the development phase has become a necessary process across many domains that have no experience with safe and secure design strategies.

Uncertainty and rapid change in the threat environment prevent a requirement-driven process that produces static design strategies. Future systems need to be designed for agility in response to context-driven changes, resilience to threat intrusions and cascading failure modes, and the ability to gracefully degrade or self-heal in response to unintended use. The knowledge of system interfaces at every level needs to persist for the life of the system and reflect the current state of operation. Methods and tools must support greenfield (new) and brownfield (existing) implementations.

This is a policy, economics, and technology problem, and today we don't effectively systems engineer models of the system and its external context. Scenarios and concept of operations used to develop the systems need to cover and persist at every level of the design and across many external context drivers. Today, we don't have a systems language that spans engineering, policy, and economic concerns.

81.4.4.3 Desired Results

The future state of systems engineering will capture system security value and risk metrics into all types of systems and all levels and phases of decision analysis. In order to make effective decisions, systems engineering methods, processes, and tools will span the engineering, economics, and policy domains. Best practices from systems engineering (technology driven) and engineering design (user experience driven) will be merged so that all systems balance solutions and external context drivers.

Systems engineering as a whole will "catch up" so that we can engineer system solutions at the speed of the threats we see today. System models will reflect all system behaviors, be adaptive, and be able to evaluate systemic effects both within the system and out to external use contexts. Continuous experimentation with system vulnerabilities and threat scenarios (red and blue teams) will become a standard part of SE at all phases of design, integration, test, and deployment.

Model-based systems knowledge of both the system and current context will always reflect the current states of the system and its history.

Threat actors and environmental threats will always be attacking, exploiting, or otherwise systemically changing the context of systems and new technologies. In the future, systems will be designed to adapt to these threats instead of reacting to them.

81.4.4.4 Obstacles

Safety and security are crosscutting concerns, and they affect system engineering both at scale and in decomposition to components. Current processes are not agile with respect to external context changes and do not keep up with the knowledge developed as the system matures in its intended and emergent uses. Knowledge capture of design is not connected to knowledge capture of experience. Static requirements, hierarchical decomposition, and a point design mentality are ingrained in the systems engineering mindset, while system uses of today are continually changing and exhibit self-adapting behaviors that are decidedly nonhierarchical. But existing mindsets and historical processes are difficult to change.

These Grand Challenges are fundamentally technology, policy, and economic issues. We do not have a "language" today that allows different domains (engineering, policy, economics, etc.) to communicate. Thus, stakeholder decisions at every level are poorly informed and driven by self-interests. Engineering is still decidedly non-holistic in its perspectives, and this is instilled at every level of engineering education.

81.4.4.5 Research Questions

The group identified a number of research questions in the discussion. Many of these reflect the need for general methods to address systemic threats and change, not just safety and security engineering processes. A general call for research that increases holism and interdisciplinary design is needed, as well as the following specific questions:

- How do we categorize security- and safety-related challenges? Structural decomposition methods no longer work.
- How do you characterize capabilities and gaps in this domain?
- Are there methods and tools that can address simultaneously the human, physical, and informational aspects of the system?
- Can we address uncertainty in the system analysis process, particularly uncertainty of external system drivers?
- Is it possible to model real-time, realistic operational environments of changing systems and changing contexts?

- How can we design governance methods for ensuring system security when the individual components are not secure, effectively moving to heterarchical system structures?
- How do you "reverse engineer" the architecture of an existing system in order to learn its interfaces and potential vulnerabilities, particularly brownfield systems (power grids, etc.)?
- How do you measure what you don't know and plan to evolve as the system scales?
- How do you expose the necessary information to monitor system security in a private way?
- How do you design a system to self-analyze failures and heal itself?
- How can systems engineering education engage the different domains at this level of design?

81.5 Conclusion

Overall, this workshop succeeded in its main aim of validating the planned workshop approach and the use of Grand Challenges to focus future SE research questions. While all of the groups took different approaches, a broad consensus emerged on the types of research questions raised, in particular:

- Ways to model complex situations as systems and use this to improve understanding
- Ways to extend SE approaches to system integration and option assessment, to relate to the scale of challenges faced in considering solutions
- Ways to consider complex emergent issues across such solutions

The idea of a *socioeconomic enterprise* discussed in Sect. 81.4.2 may provide a generic context into which many of the Grand Challenge themes could be placed setting a scope which covers all aspects of its outcomes whoever they are delivered by. Such enterprises must consider the balance of finite resources and tradeoffs across the full scope, how to set the necessary level of human and technical integration, and the need to remain viable within environmental factors and possible threats. All decisions in such an enterprise must consider what is acceptable within the social context in which they sit or working to change the views within such a context.

Such a way of looking at national and global challenges might help to focus on issues for which SE can provide answers and to create research challenges to help drive the ongoing transformation of SE as a professional discipline. In our future work in workshops 2 and 3, we will explore these possibilities.

Acknowledgments The authors wish to thank INCOSE, the Systems Engineering Research Center (SERC), and Johns Hopkins University Applied Physics Laboratory for their support of this project. Particular thanks go to the keynote speakers: Kristen Baldwin, Dennis Buede, and

Rick Adcock who provided inspiration to this first workshop. Finally, thanks to the attendees who provided the content for this report: Food and clean water (Ellie Gianni, John Gill, Susan Plano-Faber, Nazanin Sharifi, and Timothy Sprock), Healthcare (Mimi Hailegiorghis, Ken Harmon, Duane Hybertson, Ronald Luman, and Conrad Tucker), Education (Dennis Buede, Jim Coolahan, Don Gelosh, Mike Pafford, and Shamsmaz Virani), and Security (Darryl Farber, David Flanigan, Scott Lucero, Laura Rogers, Loren Walker, Joseph Williams, and Harry Winsor).

References

1. Kennedy JF (1962) Speaking at rice university, Sept 12
2. Hilbert D (1900) Mathematische Probleme. Göttinger Nachrichten:253–297
3. Moore G (1965) Cramming more components onto integrated circuits. Electro Mag 38 (8):114–117
4. Varmus H et al (2003) Grand challenges in global healthscience. Science 302(5644):398–399. Bill & Melinda Gates
5. National Academy of Engineering (2008) Grand challenges for engineering. Downloaded 23 Oct 2016 http://www.engineeringchallenges.org/File.aspx?id=11574&v=ba24e2ed
6. Nesta. Longitude prize. https://longitudeprize.org/
7. XPrize Foundation. http://www.xprize.org/
8. American Academy of Social Work & Social Welfare. The grand challenges for social work. http://aaswsw.org/grand-challenges-initiative/12-challenges/
9. Kalawsky RS (2009) Grand challenges for systems engineering research: setting the agenda. In: Conference on systems engineering research, Loughborough
10. A world in motion:systems engineering vision 2025<to be completed>
11. World Economic Forum (2016) The global risks report. Downloaded 23 Oct 2016 http://www3.weforum.org/docs/GRR/WEF_GRR16.pdf
12. Wilkerson TL (2015) Grand challenges and opportunities in mathematics education research. J Res Math Educ 45(4):402–405
13. Mekonnen M, Hoekstra A (2016) Four billion people facing severe water scarcity. Science Advances 2(2):e1500323
14. BKCASE Editorial Board (2016) The guide to the systems engineering body of knowledge (SEBoK), v. 1.7. R.D. Adcock (EIC). Hoboken, The Trustees of the Stevens Institute of Technology. Accessed 1 Nov 2016. www.sebokwiki.org
15. Rittel HWJ, Webber MM (1973) Dilemmas in a general theory of planning. Policy Sci 4:155–169. Retrieved 25 April 2013

Index

A.M. Madni et al. (eds.), *Disciplinary Convergence in Systems Engineering Research*, https://doi.org/10.1007/978-3-319-62217-0