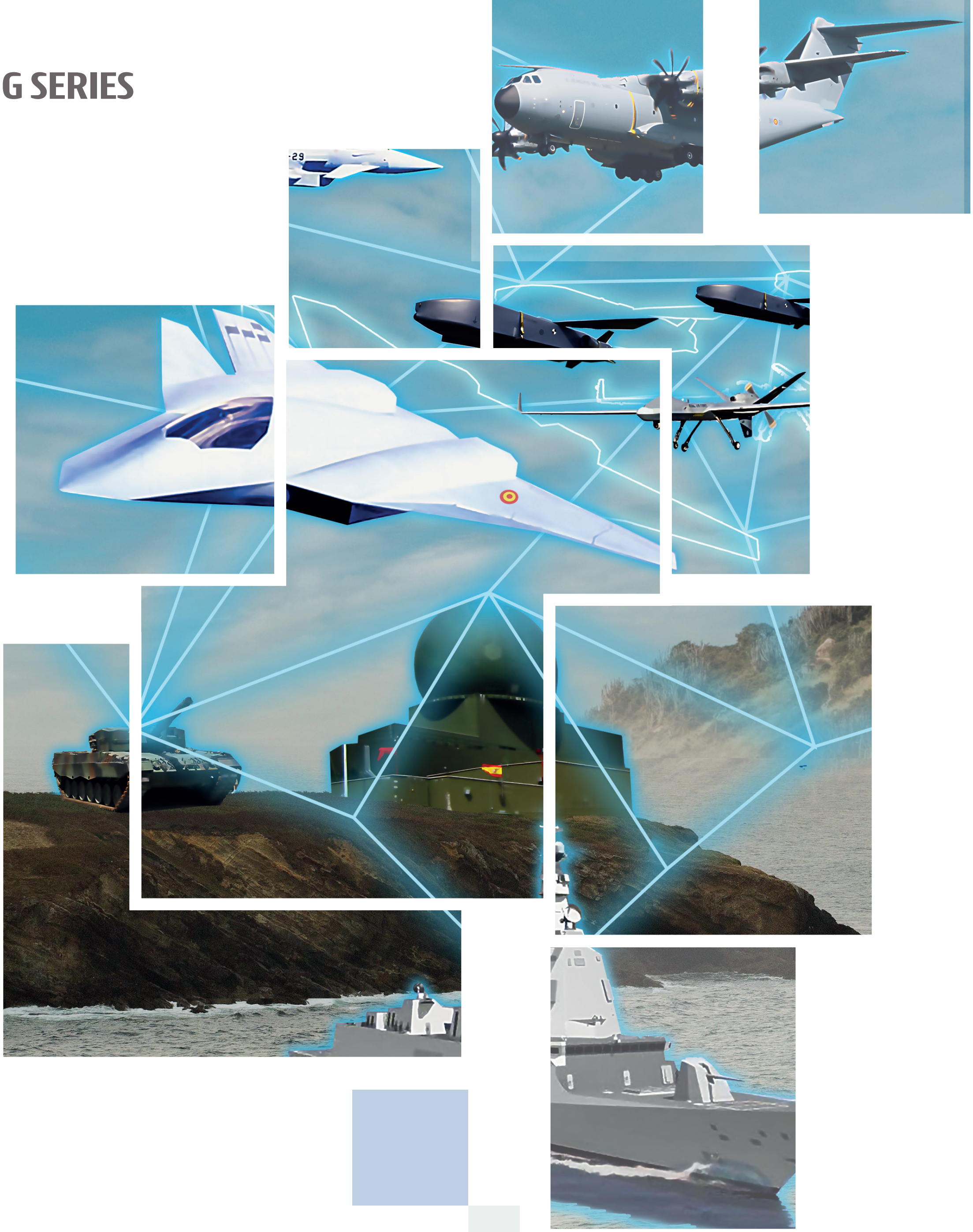
SYSTEMS ENGINEERING SERIES
Cuadernos de Isdefe



2

SYSTEMS ENGINEERING

Applied to SoS

CUADERNOS DE Isdefe

Isdefe

Systems Engineering Applied to SoS

Volume 2

Systems Engineering Series

Cuadernos de Isdefe

CUADERNOS DE
Isdefe

AUTHORS

Dra. Judith Dahmann / David Sánchez García

Dr. Michael Yokell / Dr. Alejandro Salado / Adolfo Sánchez Domínguez

Dr. James Moreland Jr / LtCol. Víctor M. Sobrino García

Matthew Gagliardi / Matthew C. Hause / Dr. James N. Martin

Dr. Polinpapilinho F. Katina / Dr. Charles B. Keating / César Heras Menor de Gaspar / Víctor Ramos del Pozo

Dr. Michael Sievers / Pablo Marticorena San José

Mark Phillips / Dr. Keith Joiner / Aurelio Fernández Sáez / Manuel Fernández Astaburuaga

Tom McDermott / Miguel Ángel Coll

José Luis de Rosario / Dr. Paul Grogan / Dr. Alessandro Golkar / Dr. Amro Farid / Dr. Young-Jun Son / Dr. Nil Egin / Dr. John Little

Isdefe

Original Title: Systems Engineering Applied to SoS

© Isdefe
C/ Beatriz de Bobadilla, 3 -28040 Madrid
www.isdefe.es

First printing: September 2025

ISBN: 978-84-09-74840-2
No comercial
Legal deposit: M-14809-2024

Publisher: Ingeniería de Sistemas para la Defensa de España SA SME MP
Coordinator: Juan Manuel García Montaño
Technical coordinator: Dr. Alejandro Salado
Editor: Dr. Alejandro Salado
Editing and style review team: Juan Manuel García Montaño, Dr. Alejandro Salado
Book design and layout: Iliana Aguilar Jiménez
Video recording and production: Favorit Comunicación

Printing by Byprint Madrid
Printed in Spain - *Impreso en España*

The opinions contained in this book are the sole responsibility of the undersigned authors. They are not intended to reflect the opinions or point of view of Isdefe as a company.
The editing team has made every effort to obtain the appropriate permissions for all material reproduced in this book. If there is an omission, we ask that you send us a written request to correct the error.

We would like to express our gratitude to the following individuals and organizations for making this edition of the second Systems Engineering in the 21st Century notebook possible:

To MG (R) Jesús Gómez Pardo prologue writer of the work.
To our technical coordinator Dr. Alejandro Salado for his constant direction and ability to solve the challenges faced.
To Rocío Manjón Pérez, collaborator and promoter of these notebooks.
Essential and permanent support to the staff of the Corporate Development and Strategy Department (especially, Lluis Vinagre, Ildefonso Vera, Natalia Garcillán and Carmen Ekame).

"The greatest danger in times of turbulence is not the turbulence; it is to act with yesterday's logic."

Peter Drucker

FOREWORD

We live in a convulsive world. The **global instability and uncertainty** of our living environment, since the fall of the Berlin Wall, are catalyzed by the **technological revolution** in which we are immersed. Emerging technologies are bringing about profound changes in the way we think and act and are accelerating the transition of our society towards the digital age.

We are facing unknown futures, enormous security and technological challenges, that we can only overcome by creating **agile structures**, in permanent adaptation to the rapid evolution of the environment, and developing **more flexible and resilient organizations**, which allow us to survive and operate in an increasingly volatile and complex environment. Our Armed Forces, which are an integral part of society, are immersed in a process of **digital transformation** that is changing the art of war. A digitized, hyper-connected battlefield, in which operations are carried out in a network by well-equipped combatants who serve as nodes in the warfighting network with a widespread use of unmanned vehicles, autonomous systems, drones, armed robots -acting individually or collaboratively- are already a reality that is substantially modifying current operational concepts.

A clear example is the conflict in Ukraine[1]. Russia's illegal invasion of Ukraine seems to have brought us to a high-intensity conventional conflict although with certain exceptions: both parties are making extensive use of specific **emerging and disruptive technologies (EDT)**, characteristic of modern conflicts; there is also a clear **confrontation in the grey zone** between Russia and the West; and, all this, against the backdrop of the nuclear threat. Operations are being carried out in the **multi-domain** which integrates both the physical -land, sea and air, including outer space- and the non-physical -cyberspace and cognitive space-, domains established by the Spanish doctrine. On the other hand, some fundamental changes in battle, associated with digitalization, have been identified, with the **massive use of drones** standing out.

From this terrible and highly attritional conflict, and focusing on **military capabilities**, we can draw the following lessons learned: that **intelligence, surveillance, reconnaissance and target acquisition** capabilities are key to superiority in operations; that it is critical to have on the battlefield **precise and long-range** fires to hit targets minimizing collateral damage; that heavy **firepower** is needed to saturate the adversary; that the battlefield is becoming digital; that it is now evident that in order for network operations to be possible, we need to ensure the **interconnection of sensors and combat systems with management centers and fire -or effects- producing elements**; and, finally, that in order to ensure this interconnection, it is essential to have **robust, redundant and highly mobile** command and control systems, as well as **electronic warfare capabilities** to operate in the electromagnetic spectrum with the necessary superiority, to guarantee operability in degraded environments and freedom of action in the five operational domains.

Ukraine has also highlighted how a combat situation spurs ingenuity and the ability to innovate and how the threat changes and evolves at a dizzying pace. That is why it is important that, in order to provide our Armed Forces with these new military capabilities, the **defense industry** is able to manufacture, in a short time, the appropriate weapons systems to face the changing threats. To this end, it will have to implement **agile development methodologies**, using the possibilities of digitalization, in the development and maturation of the **EDT** that are bursting onto the battlefield.

In this sense, our Armed Forces must have powerful **electronic warfare systems** prepared for navigation warfare **(NAVWAR)** so that the superiority of the PNT (position, navigation and timing) information can be ensured, protecting our own navigation systems and degrading the adversary's PNT information. The development of **anti-drone technologies** has become an absolute priority. To guarantee the safety of any operation, a wide range of technologies must be available for the location, identification and monitoring of the threat, as well as for its neutralization, either with soft kill means

[1] "Conclusiones iniciales de la guerra en Ucrania". Centro Conjunto de Desarrollo de Conceptos. Estado Mayor Conjunto, marzo 2023.

(jamming or spoofing) or with hard kill means – destruction kinetic effectors (projectiles, rockets, or active protection systems), nets, electromagnetic pulse, laser weapons, etc. Likewise, technological development in the fields of **robotics and autonomous vehicles** is essential to automate legacy systems (*drive-by-wire*), to develop advanced driver-vehicle interfaces and technologies for UGV-UA interaction, etc. Other fundamental technologies to be developed and made available will be those associated with **industry 4.0** (AI, cyber-physical systems, machine learning, digital twins, cloud systems, Big Data, etc.) that will allow to complete the transformation of the logistics structures of the armies, evolving from reactive-preventive logistics to **predictive logistics**. Finally, the use of **AI and *Big Data*** will be a priority for the predictive maintenance of platforms, the automatic and intelligent analysis of large volumes of data from weapons system sensors and the intelligent analysis of information sources in support of decision-making. Thus, AI will be essential, among others, for the complete simulation of the battle environment, C-UAS comprehensive combat management systems, sensor fusion, precise positioning in complex environments, autonomous navigation in unstructured environments; automation algorithms development; planning of itineraries, etc.

With regard to the **financing** of these EDT, we can say that geopolitical uncertainty and instability, together with the weakening of the transatlantic link, are forcing the growth of defense investments in order to achieve European strategic autonomy and meet NATO commitments. In this regard, the Spanish government has recently published the **Industrial and Technological Plan for Security and Defense**[2] with the aim of guaranteeing Spain's security and consolidating Spain as a central and reliable member of the European Union and NATO. The plan also aims to promote a new wave of innovation and reindustrialization of companies around dual-use technologies. Industrial investments will focus on developing, manufacturing and acquiring new telecommunications and cybersecurity capabilities, as well as on the manufacture and purchase of new defense equipment.

It is clear that we are aware of the security challenges and the military capabilities needed to meet them, that we are aware of the technological challenges and industrial capabilities essential to the development of key EDT, and that we have the economic and financial tools to obtain the most technically advanced weapons systems that provide a clear operational advantage to our combatants. And it is in this procurement process that systems engineering plays a fundamental role.

Traditional systems engineering (SE) had been applied in defense since the early 1990s. In 1992, the Systems Sub-Directorate was created within the Army Logistics Support Command. Colonel Torrón introduced the SE's vocabulary into the military field, evidencing the lack of culture that we had, at the time, regarding the important role

[2] *https://www.lamoncloa.gob.es/consejodeministros/resumenes/Documents/2025/230425-plan-industrial-y-tecnologico-para-la-seguridad-y-la-defensa.pdf*

that SE plays in the life cycle of weapons systems. Isdefe responded to the challenge of filling this gap by producing **Isdefe's first series of systems engineering publications**, initiated in 1995, the famous blue books written by great specialists in the field (Blanchard, Sarabia, Aracil, ...) In 2002, with a certain level of awareness, MALE began to work on the PRISMA Program (MALE Systems Reengineering Program), with Colonel Orts as program head. In 2003, it was officially launched with the contracting of technical assistance to Isdefe for the period 2003-2005. During this period, an incredible effort was made, both by the company and by the MALE, to achieve its implementation in 2005. PRISMA was born with a double objective:

a) To adopt the NATO methodology for the systems life cycle and

b) to provide a common working methodology, to all sections of the MALE regarding the weapons systems acquisition and maintenance process.

With a unique, well-defined vocabulary, PRISMA served as a guide to the program managers and technical directors in their relations with the awarded companies and with the official quality services. In turn, it served to "educate" companies in SE. In short, it was a set of guidelines for the proper management of weapons systems' life cycle.

However, when we analyze the current reality of the battlefield and its evolution – network operations, combat cloud, cyber defense, anti-drone systems, collaborative autonomous systems, AI, predictive logistics, quantum and photonic communications, etc. – we see that the traditional concept of a weapon system is changing. Modern systems are very complex. They are made up of independent systems interconnected with each other (radars, satellites, unmanned vehicles, communications systems, etc.) and integrated into a coordinated architecture that acts more efficiently in the fulfilment of the mission. These complex systems, characterized by a distributed governance, are called **systems of systems (SoS)**, also known, depending on the application domain, as supersystems.

In the first monograph of this group of notebooks "Introduction to systems engineering in the 21st century", the new types of systems - cyber-physical systems, systems based on learning or systems with distributed governance (system of systems) - were already described in detail, for which traditional methods of systems engineering may be ineffective which is why the need to evolve and adapt traditional systems engineering methodologies to SoS was also analyzed. Therefore, we move from systems engineering to systems of systems engineering, case which is discussed in this notebook and whose content is presented in the preface.

From what has been said so far, we see that current operations in multi-domain are inconceivable without SoS. From a military point of view, SoS are relevant to:

a) ensure **interoperability**, allowing different systems -operating in different domains- to work together sharing real-time information and coordinating actions;

b) improve strategic **decision-making**, integrating multiple sources of information that provide a global view of the field of operations and greater situational awareness;

c) ensure the **resilience and adaptability**, since, if one system fails, another can replace its function, keeping the operation running;

d) promote **scalability and technological evolution** through the incorporation of new technologies, without having to redesign the entire system from scratch, which is essential in a constantly changing technological and security (threat) environment; and

e) facilitate **multinational** coordination, allowing the integration of systems from different countries, with different protocols and technologies, in joint operations with other allies.

Given the complexity of SoS and the fact that SoS Engineering methodologies are still in the development phase, it is essential to have the **right talent**. Thus, the personnel of the Army Corp of Engineers, within the Secretary of State for Defense and the Logistics Support Commands, must be trained, on an ongoing basis, in this discipline. The monograph, that the reader has in their hands, contributes to alleviating this need. Its excellent content, prepared by true international and Isdefe experts in the field, constitutes a magnificent reference for all those with responsibilities in the life cycle of SoS and for the community of systems engineers in general.

To conclude, I would like to thank Isdefe, the coordinator of the series, Lieutenant General García Montaño, and the head of the Innovation Area, Ms. Belinda Misiego, for the opportunity they have given me to prologue this monograph. It is an honor and a privilege to have participated in the writing of this second installment of the series “Systems Engineering Notebooks. Cuadernos de Isdefe”.

Jesús Carlos Gómez Pardo.
General de División (R).
Dr. Ingeniero de Armamento.

PREFACE

This monograph presents a comprehensive and structured overview of the engineering of Systems of Systems (SoS), offering a snapshot of current practices and emergent disciplines in this complex and increasingly critical domain. It is the second monograph in the "blue" series on systems engineering published by Isdefe, continuing our effort to provide technically sound and accessible resources for practitioners in government and industry.

The monograph has been conceived with a specific audience in mind: professionals in the Spanish and European defense, security, space, energy, and transport sectors who are involved in the acquisition, development, integration, or operation of SoS. Whether acting as suppliers or customers, these professionals face the multifaceted challenges of working with systems that are independently managed, distributed, evolving, and often only loosely coordinated. The content of this monograph aims to support their understanding and decision-making with a firm grounding in current practice.

Each chapter has been authored through a unique collaboration between international experts and practitioners from Isdefe, ensuring both technical rigor and practical relevance. While the chapters can be read individually, the book has been designed as a cohesive volume, with each contribution forming part of a broader narrative on SoS engineering. Every chapter serves as an introduction to a key concept or activity within SoS engineering, providing a launching point for further exploration in the technical literature.

The book begins with an introduction to SoS (Chapter 1), followed by a discussion of how traditional systems engineering must evolve to address SoS challenges (Chapter 2), using ISO 21840 as a guiding standard. Chapter 3 introduces mission engineering as a discipline that prioritizes mission outcomes over platform capabilities. Chapter 4 presents a concrete example of a mission thread, offering readers a practical application of concepts discussed earlier.

Governance, integration, and test and evaluation, core processes that require a different mindset when applied to SoS, are discussed in Chapters 5 through 7. Chapter 8 challenges the traditional notion of system lifecycle by proposing a perspective of continuous SoS evolution. The final chapter (Chapter 9) presents advanced modeling, simulation, and analysis methods, including agent-based modeling, federated approaches, and heterofunctional graph theory, among others, to address the complex interdependencies of SoS.

Throughout, we have taken care to avoid speculative visions of the future, subjective opinions about the state of the field, or promotional content unsupported by practice. The result is a monograph that is deliberately sober, grounded in current capabilities and real-world experience. Yet, it does not ignore progress: where promising developments are emerging, they are presented in a measured, factual manner.

We recognize that the maturity of organizations in SoS engineering varies widely. Some may still be developing capabilities in areas long established in traditional systems engineering. This monograph should not be read as an unreachable ideal but as evidence that structured, methodical progress is possible and already underway in some places.

We hope this monograph serves as both a reference and a source of inspiration for those working to advance SoS capabilities in their own contexts.

On behalf of the authors, the project management team, and Isdefe, I hope that you find the reading educative, enjoyable, and useful.

Dr. Alejandro Salado
The University of Arizona

TABLE OF CONTENTS

SYSTEMS ENGINEERING APPLIED TO SOS

TABLE OF CONTENTS

CHAPTER 1

Systems of Systems: An introduction

Dr. Judith Dahmann, *The MITRE Corporation (jdahmann@mitre.org)*
David Sánchez García, *Isdefe (dsanchez@isdefe.es)*

Abstract

Systems of systems (SoS) are becoming more and more prevalent, and their fundamental characteristics pose challenges for the application of systems engineering. This chapter introduces systems of systems (SoS) – their history, their distinctive features, examples, the impact of advanced technology on SoS and the future of SoS – as context for the following chapters which address SoS engineering.

Keywords

Systems of Systems, Systems Engineering

1. A HISTORY OF SYSTEMS OF SYSTEMS TO TODAY

The concept of systems of systems (SoS) has been around long before the acceptance of the systems of systems terminology. Gorod et al [1] provides a detailed walkthrough of the SoS literature, as shown in Figure 1, which has been annotated to reflect key features of the history of SoS.

The earliest references to SoS [1, 2] include Boulding's paper on general theoretical constructs [3], where he described the concept of SoS as "the arrangement of theoretical systems and constructs in a hierarchy of complexity", viewing the SoS is an "open system" that can be affected by external events, noting, however, this definition is not distinct from how traditional systems have been defined [4]. Early on, others characterized urban city planning, systems science structures, and biological systems as SoS [5-7].

Despite the growing attention to SoS, as late as 2015, there was controversy over the definition of SoS [2]:

> As might be expected in an emerging field, there is yet no precise and widely accepted definition of SoS to which the bulk of the literature conforms, making it difficult to bound the field precisely. The literature is diverse, and there are many attempts to define and characterize SoS. Several reviews have sought to achieve some convergence [8-13].

Mark Maier in the late nineties provided a seminal perspective that has grounded SoS thinking to today [14]. This is discussed in the next section

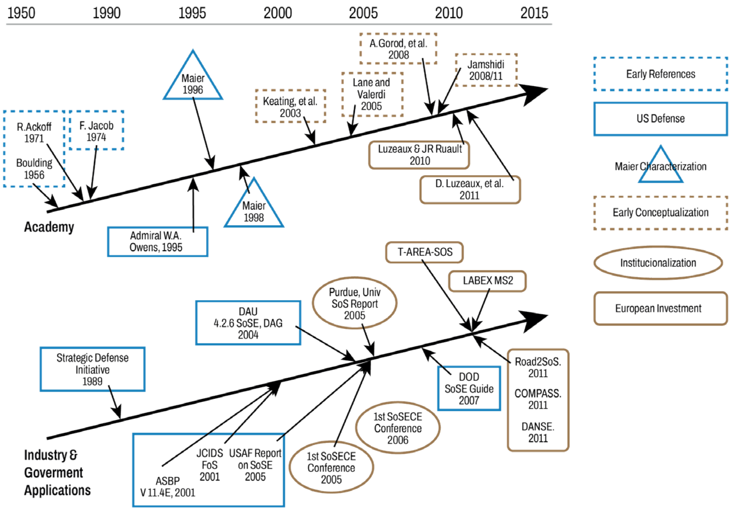


Figure 1: Modern History of Systems of Systems and Systems of Systems Engineering [1] with overlay.

The earliest examples of SoS from an engineering viewpoint come from United States (US) Department of Defense (DoD) with the US Strategic Defense Initiative starting a process of viewing defense capabilities as SoS. Admiral W.A. Owens, US Navy, introduced SoS to the military domain in "The Emerging U.S. System-of-Systems" [15]:

> *The things which give military forces their fighting capability are changing, and these changes point toward a qualitative jump in our ability to use military force effectively.*
>
> *Probably relating to the way we plan, program and budget for these things, we are more adept at seeing the individual trees than that vast forest of military capability (the system-of-systems) which the individual systems are building for our fighting forces.*
>
> *The system-of-systems depends ultimately on well-orchestrated contributions of all the military services. This assumes a common appreciation of and adherence to what we are building. Most importantly, it requires joint strategic and operational doctrine by which to organize, plan and carry out military operations.*

The US Air Force [16] and Army [17] began to take account of this perspectives, and the publication of the US DoD Guide to SoS Engineering [18], took a defense-wide engineering perspective on SoS.

In the early 2000s, institutions began to recognize systems of systems. The first annual SoS Engineering conference was held in Los Angeles in 2006, and these conferences continue through today. The International Council on Systems Engineering (INCOSE) formed a Systems of Systems working group in 2011, which provided leadership through webinars and publications including the SoS Pain Points (2013) [19], SoSE Primer (2018) [20] and Guide to SoS Standards (2020) [21]. SoS was included as a knowledge area in the SE Body of Knowledge in the first release in 2012.

Traditionally, SoS have been viewed as applying primarily to defense, largely centered on the United States. However, a meta-analysis of 168 IEEE papers published between 2020 and 2023 on SoS indicates that this is no longer the case [22]. As is shown in Table 1, SoS applications in these papers address a wide range of domains and less than a fifth of the papers address defense applications. Furthermore, the meta-analysis found that the papers were authored by authors of 29 countries, showing geographically widespread interest in SoS.

Application Areas	#	%
Defense	31	18.5%
Transportation	22	13.1%
Health	10	6.0%
IOT/CPS	8	4.8%
Energy	7	4.2%
Emergency/Crisis Mgt	6	3.6%
Space	4	2.4%
Search and Rescue	4	2.4%
Education	3	1.8%
Environment	3	1.8%
Remainder (< 3)	70	41.7%

Table 1: Domain application areas addressed by papers.

In Europe, the European Commission initiated an SoS research initiative in 2011 [23]:

> *"(ICT-2011.3.3) with an objective to increase the competitiveness of European industry and enable Europe to master and shape future developments in ICT (Information and Communication Technologies) so that the future demands of its society and economy will be met. Competitiveness means, in this context, that Europe will be global leaders in SoSE, which will lead to greater Return on Investment (ROI) for European industry, greater innovation within the technical systems community in government, industry, and academia, and long-term economic sustainability of, and through, engineering of large complex systems."*

There were four major projects in this initiative: T-AREA-SoS (Trans-Atlantic Research and Education Agenda in Systems of Systems), with the objective to develop and deliver to the European Commission a Strategic Research Agenda in Systems of Systems Engineering (SoSE); Designing for Adaptability and evolutioN in System of systems Engineering (DANSE), focused on the development of new approaches to the design and management of the operation of SoS; Comprehensive Modelling for Advanced Systems of Systems (COMPASS), which developed new modelling technology for advanced software; and ROAD2SoS, which focused on the development of strategic research and engineering roadmaps in Systems of Systems Engineering and related case studies. These projects seeded interest in SoS across European universities and industry.

These actions have shaped the direction and growth of SoS. In one of the earliest documents to address engineering applied to systems of systems, an "SoS is defined as a set or arrangement of systems that results when independent and useful systems are integrated into a larger system that *delivers unique capabilities*" [24]. This document notes that individual systems and SoS conform to the accepted definition of a system in that each consists of parts, relationships, and a whole that is greater than the sum of the parts; however, although an SoS is a system, not all systems are SoS. It also makes clear that simply having multiple systems does not make a SoS; a SoS includes the fact that these multiple systems result in a new capability not originally anticipated and/or intended by the systems alone.

In 2019, the International Standards Organization (ISO) adopted the first standards for SoS [25]. An ISO SoS Standards study group [26] recognized the increased attention to SoS and the value to standards to the maturation of SoSE. These include a definition [27] of SoS and constituent systems:

- **System of Systems (SoS)** - Set of systems or system elements that interact to provide a unique capability that none of the constituent systems can accomplish on its own.
- **Constituent Systems** - Constituent systems can be part of one or more SoS. Each constituent is a useful system by itself, having its own development, management goals and resources, but interacts within the SoS to provide the unique capability of the SoS.

This established the first widely accepted SoS definition and provided grounding for today's SoS community. Note that the definition of system in ISO is essential to interpret the definition of SoS. This is because a system (which form SoS) must be useful by itself and decides to interact within the SoS. For example, a battery powering an engine in a car would not be considered a constituent system in this context, but an element of the car.

Transportation	Multiple transportation assets and services work together to provide integrated transportation capabilities.	Air Traffic Management: SES: Single European Sky, SESAR activities and the road to Digital European Sky [28], SkyNex iTEC [29], next generation Air Traffic Control System. Rail network: Single European Rail Area, European Rail Traffic Management System (ERTMS) and other TEN-T [30] projects.
Energy	Smart grid, smart houses, and integrated production/ consumption provide energy management services.	European Network of Transmission System Operators for Electricity (ENTSO-E) vision on European Power System [31].
Health Care	Regional facilities management, emergency services, and personal health management.	E-health initiative: connecting health systems in Europe [32].
Defense	Military missions such as missile defense, networked sensors, command and control.	Joint or Cooperating ISR systems composed of Land, Air, Space, Naval ISR Systems collaborating for Joint ISR at national or international level [33]. AFSC: Alliance Future Surveillance and Control [34]. NGWS/FCAS: New Generation Weapon System / Future Combat Air System [35]. Europe's FCAS will see next-generation manned jets flying alongside unmanned remotely piloted carriers of varying sizes. These assets will be part of a fully networked 'system of systems'. FMN [36]: Federated Mission Networking is a capability aiming to support command and control and decision-making in future operations through improved information-sharing.
Telecommunications	Telecommunications systems provide telecommunications services to multiple domains.	GNSS/Galileo [37]: Galileo is Europe's Global Navigation Satellite System (GNSS) provides positioning and timing information used in smartphones, and in applications such as railways, aviation, agriculture, maritime and more. Smart Cities [38] with initiatives like ICC [39]: The Intelligent Cities Challenge (ICC) is one of the European Commission's largest initiatives supporting European cities in their green and digital transitions. The IRIS2 [40] Satellite Constellation is the European Union's third flagship, addressing long-term challenges of EU's security, safety and resilience by offering enhanced connectivity services to governmental users.
Natural Resource Management	Global environment, regional water resources, forestry, and recreational resources.	Copernicus [41] Copernicus is the Earth observation component of the European Union's Space programme, looking at our planet and its environment to benefit all European citizens. It offers information services that draw from satellite Earth Observation and in-situ (non-space) data.
Security and Disaster Response	Responses to disaster events including forest fires, floods, terrorist attacks, border control.	European Civil Protection Mechanism and the Emergency Response Coordination Center (ERCC) [42]. European Crisis Management Mechanism [43]. The EU might be exposed to a variety of crises and disasters (such as those caused by climate change, health threats, terrorist and cyber-attacks, political instability and violent conflict, failures in critical infrastructure) and should be capable to respond fast and appropriately. EuroSur/FRONTEX [44] the European Border Surveillance system (EUROSUR) is a framework for information exchange and cooperation between Member States and Frontex to improve situational awareness and increase reaction capability at the external borders.
Science	Astronomy, computing centres, research centers.	Astronomy: Distributed telescopes like LOFAR [45] (Low Frequency Array), European VLBI network [46], BOOTES [47], the first worldwide network of robotic telescopes. CERN. The accelerator complex at CERN [48] is a succession of machines with increasingly higher energies. EuroHPC JU [49], is a joint initiative between the EU, European countries and private partners to develop a World Class Supercomputing Ecosystem in Europe.

Table 2: Examples of systems that could be potentially considered Systems of Systems.

2. SYSTEMS OF SYSTEMS CHARACTERISTICS

2.1. Distinguishing Systems of Systems

As noted above, Maier's characterization of SoS is foundational to our understanding of SoS. Maier [14] presented five SoS characteristics that can be observed in most SoS, although not all of them are necessary conditions for something to be considered an SoS:

- Operational independence of constituent systems,
- Managerial independence of constituent systems,
- Geographical distribution,
- Emergent behavior, and
- Evolutionary development processes.

Of these, Maier identified operational independence and managerial independence as the two principal distinguishing characteristics for applying the term 'systems of systems.' He argues that **a system without at least one of these two characteristics is not considered an SoS** regardless of the complexity or geographic distribution of its components.

In terms of emergent behavior, "the concept of emergence refers to phenomena that occur on a system level without being present at the level of elements in the system" [50]. For systems of systems this means that there may be results or behavior not predicted by the individual constituent systems. As is discussed in the Systems Engineering Body of Knowledge [51]:

> In the Maier characterization, emergence is noted as a common characteristic of SoS particularly in SoS composed of multiple large existing systems, based on the challenge (in time and resources) of subjecting all possible logical threads across the myriad functions, capabilities, and data of the systems in an SoS... [So] there are risks associated with unexpected or unintended behavior resulting from combining systems that have individually complex behavior. These become serious in cases which safety, for example, is threatened through unintended interactions among the functions provided by multiple constituent systems in a SoS.

Finally, in terms of geographical distribution, this is just something that most of the existing SoS share but it is more a consequence of how most systems of systems are deployed than a requirement for being a SoS.

2.2. Systems of Systems Pain Points

The characteristics presented in the previous section are important because they provide the keys to the particular challenges systems engineers face when applying systems engineering to SoS. These challenges are reflected in the SoS Pain Points (Figure 2), which were identified by the INCOSE SoS working group and which have been particularly useful in understanding SoS.

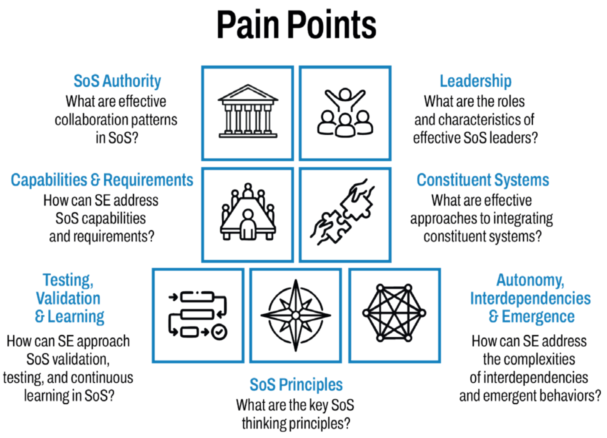

Pain Points
SoS Authority
What are effective collaboration patterns in SoS?
Leadership
What are the roles and characteristics of effective SoS leaders?
Capabilities & Requirements
How can SE address SoS capabilities and requirements?
Constituent Systems
What are effective approaches to integrating constituent systems?
Testing, Validation & Learning
How can SE approach SoS validation, testing, and continuous learning in SoS?
Autonomy, Interdependencies & Emergence
How can SE address the complexities of interdependencies and emergent behaviors?
SoS Principles
What are the key SoS thinking principles?

Figure 2: System of Systems Pain Points [40].

These pain points are briefly described in Table 3. They are presented in the INCOSE SE Handbook [52], with particular attention to the impact they have on the application of systems engineering to SoS.

SoS Authorities	In an SoS, each constituent system has an ‘owner’, stakeholders, users, business processes, and development approach, departing from traditional top-down authority over the development and operation of the SoS.
Leadership	The lack of common authority across an SoS means that decisions for the SoS rely less on traditional command and control and more on influence and persuasion, which can be accomplished in several ways, one of them being leadership.
Constituent Systems’ Perspectives	Most SoS are composed of pre-existing constituent systems , each bringing with them their own perspective, which may or may not align with perspectives of the other constituent of the SoS.
Capabilities and Requirements	Traditionally a system has a coherent set of user capabilities. SoS are comprised of multiple independent systems each with their own capabilities, which when combined may or may not provide coherent SoS capabilities.
Autonomy, Interdependencies and Emergence	The independence of constituent systems means that a constituent system may change independently of the SoS, and even leave the SoS, and impact other constituent and the SoS in unexpected or unpredictable ways.
Testing, Validation, and Learning	Since SoS are composed of independent constituent systems, this poses challenges in conducting end-to-end SoS testing as is typically done with systems.
SoS Principles	Work is needed to identify and articulate the cross-cutting principles that apply to SoS in general and to developing working examples of the application of these principles.

Table 3. SoS Pain Points.

2.3. Systems of Systems taxonomy

The defining characterization of SoS as lacking a top-level authority is central to the challenges posed by SoS. It is this characteristic that provides the driver for the most widely accepted taxonomy for SoS. This taxonomy has been adopted in ISO/IEC/IEEE 21841. As described in the SoS knowledge area of the SE Body of Knowledge [51], in those situations where the SoS is recognized and treated as a system in its right, an SoS can be described as one of four types [14,53,54]:

- **Directed:** The SoS is created and managed to fulfill specific purposes, and the constituent systems are subordinated to the SoS. The constituent systems maintain an ability to operate independently; however, their normal operational mode is subordinated to the central managed purpose.
- **Acknowledged:** The SoS has recognized objectives, a designated manager, and resources for the SoS; however, the constituent systems retain their independent ownership, objectives, funding, and development and sustainment approaches. Changes in the systems are based on cooperative agreements between the SoS and the system.
- **Collaborative:** The component systems interact more or less voluntarily to fulfill agreed upon central purposes. The central players collectively decide how to provide or deny service, thereby providing some means of enforcing and maintaining standards.
- **Virtual:** The SoS lacks a central management authority and a centrally agreed upon purpose for the SoS. Large-scale behavior emerges —and may be desirable— but this type of SoS must rely on relatively invisible mechanisms to maintain it.

Figure 3 illustrates these four types.

In reality, most actual SoS are a combination of these types.

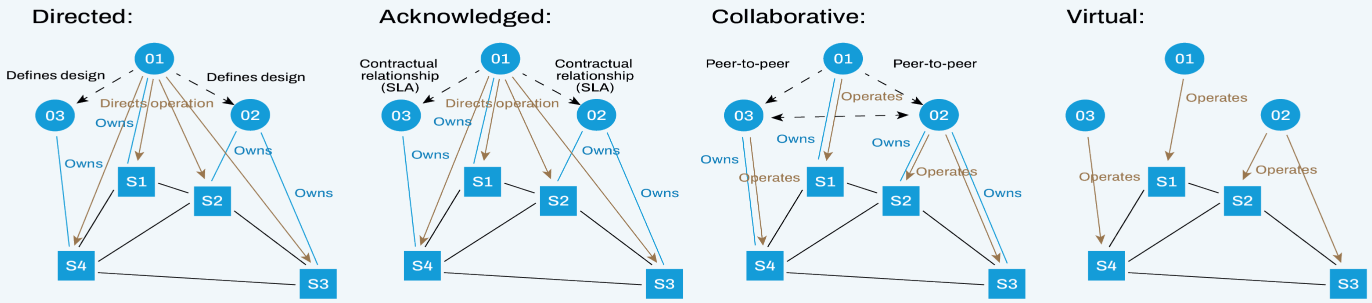


Figure 3: SoS Types [55].

2.4. SoS scale, scope and complexity

SoS can range in scale and scope, as shown in Figure 4, which draws examples from the 2011 European Commission SoS research initiative discussed above.

On the one hand, SoS may be composed of purely technical systems with the integration of heterogeneous systems developed independently into a composite capability. The figure illustrates the integration of independently developed audiovisual capabilities into an audiovisual home entertainment system of systems.

There are broader socio-technical SoS in areas such as disaster response, which include interactions among various elements in responding to a disaster including fire, public safety, volunteer organizations and others.

Finally, SoS may address enterprise-wide issues such as counterfeiting in US defense [56], which incorporate systems, organizations, policies, and competing efforts.

In each case, there are independent elements –some technical, some organizational– which compose the SoS.

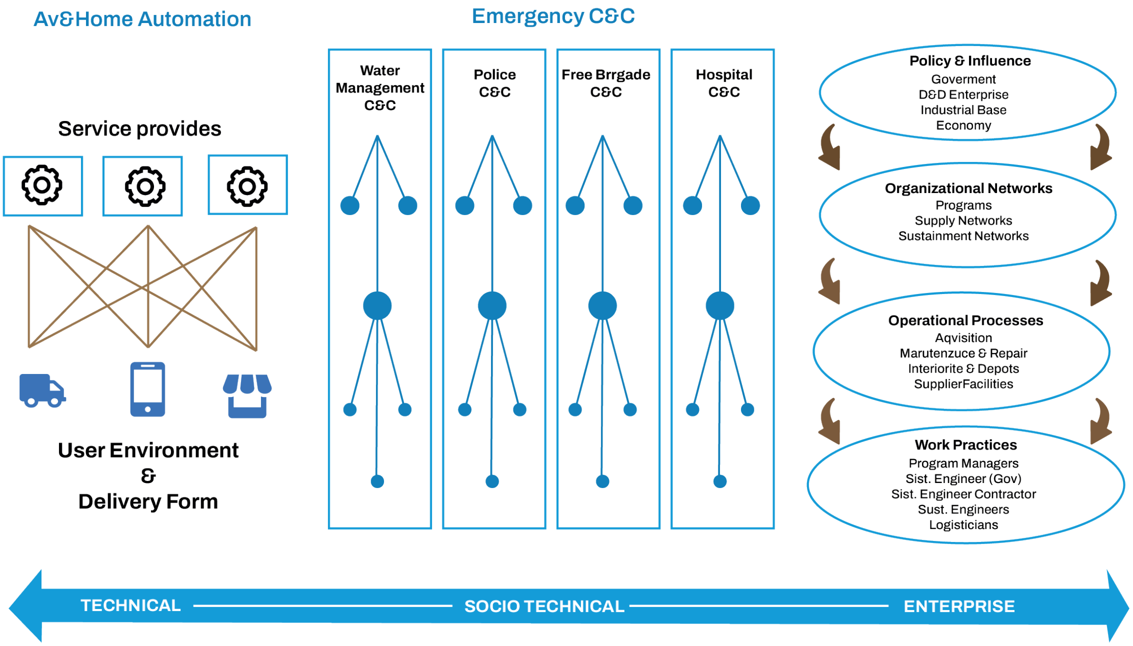


Figure 4: Scale and scope of SoS [57].

As SoS increase in scope and scale and increasingly incorporate non-technical elements, as noted in Maier characteristic of SoS emergence, SoS are particularly susceptible to complexity [58], which is defined according to the SEBOK [59] as:

> "A measure of how difficult it is to understand how a system will behave or to predict the consequences of changing it. It occurs when there is no simple relationship between what an individual element does and what the system as a whole will do, and when the system includes some element of adaptation or problem solving to achieve its goals in different situations. It can be affected by objective attributes of a system such as by the number, types of and diversity of system elements and relationships, or by the subjective perceptions of system observers due to their experience, knowledge, training, or other sociopolitical considerations."

Using work on complexity by INCOSE as the frame of reference [60], SoS by their very nature can be shown to exhibit many dimensions of complexity, and the guiding principles to complexity thinking [61] can be applied to address complexity in SoS.

3. IMPACT OF ADVANCED TECHNOLOGIES ON SYSTEMS OF SYSTEMS[1]

It is recognized that current technological advances are impacting systems in a variety of ways. The question addressed here is how technological advances are affecting SoS. Figure 5 shows the set of technologies that are having an impact on systems of systems.

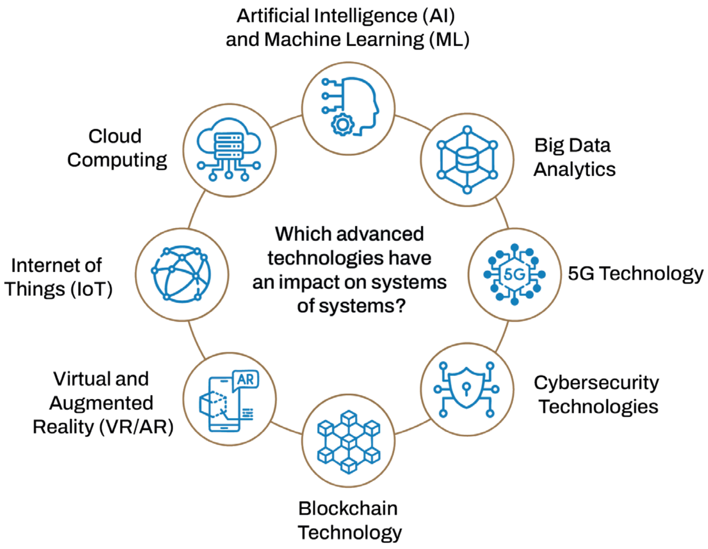

Artificial Intelligence (AI) and Machine Learning (ML)
Big Data Analytics
5G Technology
5G
Cybersecurity Technologies
Blockchain Technology
Virtual and Augmented Reality (VR/AR)
AR
Internet of Things (IoT)
Cloud Computing
Which advanced technologies have an impact on systems of systems?

Figure 5. Technologies affecting systems of systems.

Opportunities to apply **artificial intelligence (AI) and machine learning (ML)** are growing rapidly particularly with the advent of access to large language models. For SoS, these technologies provide opportunities to automate complex tasks, to optimize system performance, and to make predictive analyses. With AI and ML, SoS can learn and evolve over time, and they enable SoS characteristics of evolutionary development and operational independence. These potential advantages come with the challenges of the need for high computational power and specialized knowledge and added unpredictability. In terms of SoS pain points, AI/ML address SoS authority challenges with opportunities for intelligent decision making and new capability to address capability and requirements challenges by providing a means to learn and adapt to changes. On the other hand, the opaque nature of some AI/ML models, often referred to as the "black box" problem can make it difficult to understand and predict system behavior.

Big data analytics offer the opportunity to handle and analyze large volumes of data from different systems, providing insights for SoS design and operation and enabling data-driven decision making and opportunities for overall efficiency and performance. These potential opportunities come with the challenges of handling the large volume, velocity, and variety of data including data management, storage, and analysis and data quality and integrity. From the perspective of SoS pain points, big data analytics can provide insights into behavior and performance of systems and opportunities for better integration and management of constituent systems. But the lack of cross cutting SoS authority can pose data ownership and privacy issues and given the diversity of constituent system perspectives the need to integrate and manage large volumes of data from different systems.

[1] *Material in this section is based on a panel presentation at the IEEE SoS Conference INCOSE Panel presentation in June 2024 in Tacoma, Washington (USA) using MChat, the Microsoft Azure OpenAI GPT-4 32K LLM model available at MITRE.*

5G technology provides high-speed, reliable, and low latency communications, which are crucial in many system of systems, and as a result can enable enhanced communication and data sharing and real-time data sharing and collaboration. This comes with the challenge of significant investment in infrastructure and new challenges of security and privacy of data. In terms of the autonomy, interdependencies and emergence SoS pain point, the advantage of high-speed connectivity/low latency real-time data sharing and communication among systems comes with the challenge of increased complexity and potential for interference between systems.

Cybersecurity technologies can help protect systems of systems from potential threats and attacks, ensure the security and integrity of data, and protect the SoS from potential threats and attacks. Yet, SoS continue to face security challenges of each constituent system and their impact on the overall system of systems. Given the lack of SoS authority, SoS face the challenges of ensuring the security and integrity of data across systems and building trust among different systems and their authorities including the need for coordination and compliance with different security protocols and standards across systems.

Blockchain can ensure transparency and security in transactions and provide the opportunity to create secure and transparent systems, enhance trust among different systems and stakeholders, and ensure data integrity and security. However, this requires a deep understanding of the technology.

Virtual and augmented reality technologies allow for the simulation of real-world scenarios. This addresses the testing, validation, and learning SoS pain point. It employs immersive and interactive platforms for testing and validating systems, creating the opportunity to establish secure and transparent systems and enhance trust among different systems and stakeholders. However, these technologies can be technically challenging, requiring specialized knowledge, and areprone to unrealistic expectations.

Cloud computing allows for the integration of various systems on a common platform that enables interaction and collaboration, operational independence, and geographical distribution. This enables enhanced integration and interoperability of systems and provides a scalable and flexible platform that can accommodate the evolving needs of the SoS. The challenges of data security and privacy concerns and of managing and integrating diverse systems remain, as data control becomes distributed across different cloud services and issues with data sovereignty, as well as compliance, given the lack of a centralized SoS authority.

Internet of Things (IoT) technologies provide interconnectivity between different systems, allowing them to communicate and share data. These technologies enable geographic distribution, real-time data collection and communication, efficient system design and operation, and improved adaptability to changing conditions. At the same time, they face challenges of data overload, management and analysis of distributed data, and security of IoT devices, given the complexity resulting from the sheer number of devices and systems to be managed and coordinated, which introduce new security vulnerabilities.

4. CONCLUSIONS

It is safe to say that systems of systems are here to stay and if anything, are becoming recognized as a class of systems warranting special consideration. With the advent of modularity in systems design, what may have been considered single technical systems, now exhibit many of the SoS characteristics, as fewer systems are developed completely from the ground up. System components developed by independent organizations are integrated as independent constituent systems of new systems, bringing with them the benefits of reuse along with some of the challenges of SoS. As systems engineering continues to apply to larger socio-technical enterprises, these organizational systems of systems are part and parcel of the future systems landscape and part of the remit of systems engineering.

This section has laid the groundwork for the SoS engineering topics addressed in the remainder of this monograph. The history of SoS has been reviewed along with the critical characteristics that distinguish this class of system and how these characteristics drive SoS complexity. The chapter has shown how SoS are found across a wide range of domains and are the topic of research globally. Advanced technologies are key drivers for SoS which are going to be increasingly important in the future. These provide important context for understanding the challenges and approaches to systems of systems engineering.

REFERENCES

1. Gorod, Alex, Brian Sauser, and John Boardman. System-of-Systems Engineering Management: A Review of Modern History and a Path Forward. 2015 IEEE SYSTEMS JOURNAL, VOL. 2, NO. 4, DECEMBER 2008

2. Neilsen, Claus Ballegaard, Peter Gorm Larsen, John Fitzgerald, Jim Woodcock, and Jan Peleska. 2015. Systems of systems engineering: Basic concepts, model-based techniques, and research directions. ACM Comput. Surv. 48, 2, Article 18 (September 2015)

3. Boulding, Kenneth. 1956. General systems theory—The skeleton of science. Management Science 2, 3 (April 1956).

4. Von Bertalanffy, Ludwig. 1969. General Systems Theory. New York: George Braziller.

5. Berry, Brian J. L.. 1964. Cites as systems within system of cites. Papers and Proceedings of the Regional Science Association 13, 1 (Jan. 1964), 149–163.

6. Ackoff, Russell L.. 1971. Towards a system of systems concept. Management Science 17, 11 (July 1971), 661–671.

7. Jacob Francois. 1974. The Logic of Living Systems: A History of Heredity. Allen Lane.

8. Keating, C., R. Rogers, R. Unal, et al. 2003. Systems of systems engineering. Eng. Management J. 15-3: 36–45.

9. Jamshidi, M. 2005. System of systems engineering – A definition. Piscataway, NJ: IEEE SMC.

10. Abeer Sharawi, Serge N. Sala-Diakanda, Sergio Quijada, Nabeel Yousef, Luis Rabelo, and Jose Sepulveda. 2006. A distributed simulation approach for modeling and analyzing system of systems. In 2006 Winter Simulation Conference.

11. Lane, J. A. and R. Valerdi, "Synthesizing SoS concepts for use in cost estimation," presented at the IEEE Conf. Syst., Man, Cybern. Waikoloa, HI, 2005.

12. Gorod, A., B. Sauser, and J. Boardman. 2008. System-of-Systems Engineering Management: A Review of Modern History and a Path Forward. IEEE Systems Journal, 2-4: 484-99. http://ieeexplore.ieee.org/xpl/login.jsp?tp=&arnumber=4682611&url=http%3A%2F%2Fieeexplore.ieee.org%2Fxpls%2Fabs_all.jsp%3Farnumber%3D4682611.

13. Ed. Jamshidi, M. 2008. System of System Engineering – Innovations for the 21st Century. Hoboken, NJ: Wiley.

14. Maier, M. 1998. Architecting principles of systems-of-systems. 6th Ann. Int. Symp. Int. Council Syst. Eng. Boston, MA.

15. Owens, William A.. "The Emerging US Systems-of-Systems" in Strategic Forum, Institute for National Strategic Studies, The National Defense University, Number 63, February 1996.

16. United States Air Force Scientific Advisory Board. Report on System-of-Systems Engineering for Air Force Capability Development Executive Summary and Annotated Brief. SAB-TR-05-04. July 2005.

17. Department of Defense. 2001. Army software blocking policy: Version 11.4E.

18. DoD. 2009. System of systems, systems engineering guide: Considerations for systems engineering in system of systems environment. U. S. Department of Defense.

19. J. Dahmann. 2014. Systems of systems pain points. In INCOSE International Symposium on Systems Engineering 2014.

20. INCOSE. 2018. Systems of Systems Primer. INCOSE-TP-2018-003-01.0.

21. INCOSE. 2018. Guide to SoS Standards

22. Dahmann, Judith. Current Landscape of Systems of Systems Engineering, IEEE SoSE Conference, Tacoma, WA, June 2024.

23. Henson, S.A, M.J.D. Henshaw, V. Barot, C.E. Siemieniuch, M.A. Sinclair M. Jamshidi H. Dogan, S.L. Lim, C. Ncube D. DeLaurentis, Towards a Systems of Systems Engineering EU Strategic Research Agenda. Proc. of the 2013 8th International Conference on System of Systems Engineering, Maui, Hawaii, USA - June 2-6, 2013

24. DAU, "Defense acquisition guidebook," 2004.

25. ISO/IEC/IEEE 21839 (ISO, 2019)

26. ISO SoS Standards study group report (ISO, 2016)

27. Quick Reference Guide to SoS Standards, INCOSE, 2020.

28. https://www.sesarju.eu/MasterPlan2025 Accessed March 2025

29. https://www.itecskynex.com/ Accessed March 2025

30. https://transport.ec.europa.eu/transport-themes/infrastructure-and-investment/trans-european-transport-network-ten-t_en Accessed March 2025

31. https://vision2030.entsoe.eu/ Accessed March 2025 Accessed March 2025

32. https://health.ec.europa.eu/document/download/3dc615dc-e94c-4691-b655-a2877a6aa57c_en?filename=2016_ehealthleaflet_vertical_en.pdf Accessed March 2025

33. https://www.nato.int/cps/fr/natohq/topics_111830.htm?selectedLocale=en Accessed March 2025

34. https://www.nato.int/nato_static_fl2014/assets/pdf/2020/7/pdf/200701-Factsheet_Alliance_Future_Surveil-1.pdf Accessed March 2025

35. https://www.airbus.com/en/newsroom/stories/2023-11-future-combat-air-system-fcas-enter-the-internet-of-military-things Accessed March 2025

36. https://www.act.nato.int/activities/federated-mission-networking/ Accessed March 2025

37. https://defence-industry-space.ec.europa.eu/eu-space/galileo-satellite-navigation_en Accessed March 2025

38. https://commission.europa.eu/eu-regional-and-urban-development/topics/cities-and-urban-development/city-initiatives/smart-cities_en Accessed March 2025

39. https://www.intelligentcitieschallenge.eu/about-icc Accessed March 2025

40. https://defence-industry-space.ec.europa.eu/eu-space/iris2-secure-connectivity_en Accessed March 2025https://civil-protection-humanitarian-aid.ec.europa.eu/what/civil-protection/eu-civil-protection-mechanism_en Accessed March 2025

41. https://www.copernicus.eu/en Accessed March 2025

42. https://civil-protection-humanitarian-aid.ec.europa.eu/what/civil-protection/eu-civil-protection-mechanism_en Accessed March 2025 Accessed March 2025

43. https://joint-research-centre.ec.europa.eu/jrc-science-and-knowledge-activities/crisis-management_en Accessed March 2025 Accessed March 2025

44. https://home-affairs.ec.europa.eu/policies/schengen-borders-and-visa/border-crossing/eurosur_en Accessed March 2025

45. https://www.glowconsortium.de/index.php/en/lofar-about-new Accessed March 2025

46. https://evlbi.org/ Accessed March 2025

47. https://bootesnetwork.com/ Accessed March 2025

48. https://home.cern/science/accelerators Accessed March 2025

49. https://eurohpc-ju.europa.eu/index_en Accessed March 2025

50. Axelsson. Jacob. What Systems Engineers Should Know About Emergence. First published: 26 September 2022; https://doi.org/10.1002/iis2.12982

51. Systems Engineering Body of Knowledge (SEBOK), https://sebokwiki.org/wiki/Guide_to_the_Systems_Engineering_Body_of_Knowledge_(SEBoK), Accessed March 2025

52. INCOSE. 2015. INCOSE Systems Engineering Handbook: A Guide for System Life Cycle Processes and Activities, Version 3.2.2. (4th ed.). Wiley.

53. Dahmann, Judith and Kristen Baldwin. 2008. Understanding the current state of US defense systems of systems and the implications for systems engineering. In IEEE Systems Conference. IEEE.

54. ISO 21841, Taxonomy of systems of systems, 2019.

55. Henshaw, Michael. Emerging Strategic Research and Education Agenda in SoS: Trans-Atlantic Research and Education Agenda in Systems of Systems. NDIA presentation, April, 2013.

56. Bodner, Douglas. Mitigating Counterfeit Part Intrusions with Enterprise Simulation. Procedia Computer Science 61 (2015) 233 – 239

57. Dahmann, Judith. Systems of Systems Characterization and Types. NATO Publication Ref NBR EN-SCI-276-01.2015.

58. Dahmann, Judith and Dan DeLaurentis, Unique Challenges in System of Systems Analysis, Architecting, and Engineering. In Systems Engineering for the Digital Age. Wiley. 2024.

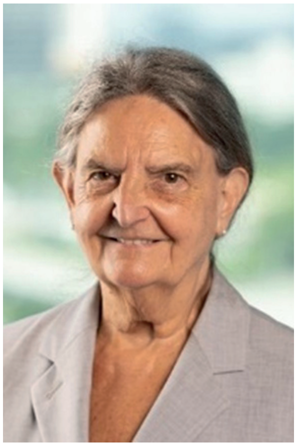
59. SEBOK: Emerging Topics: SoS and Complexity. https://sebokwiki.org/wiki/Guide_to_the_Systems_Engineering_Body_of_Knowledge_(SEBoK) Accessed March 2025

60. Watson et al. “Appreciative Methods Applied to the Assessment of Complex Systems”, 29th INCOSE international Symposium, Orlando, Florida, July 20-25, 2019

61. INCOSE Complexity Primer; INCOSE, 2016

BIOGRAPHIES

DR. JUDITH DAHMANN



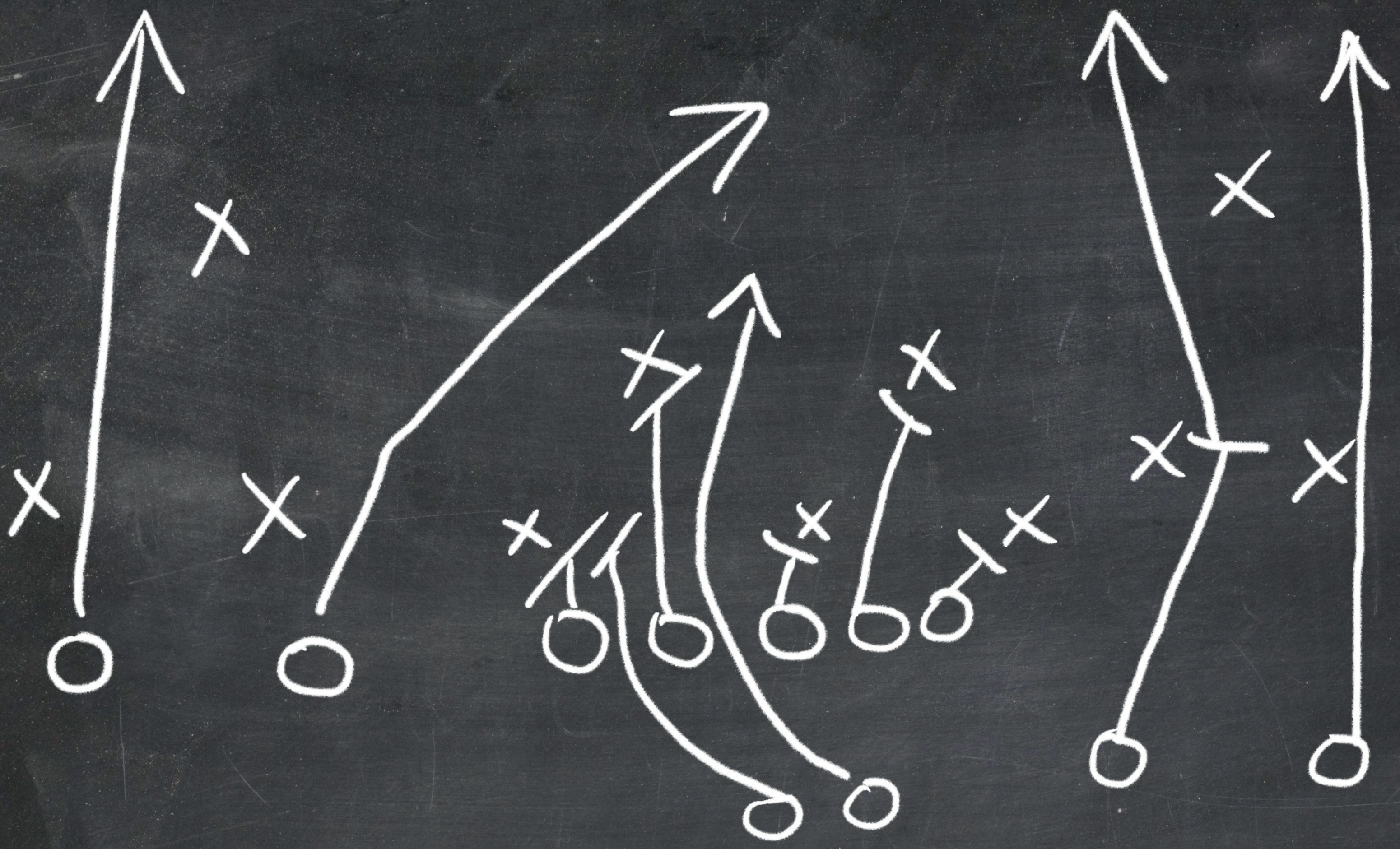
Dr. Judith Dahmann is a MITRE Fellow at the MITRE Corporation and the MITRE lead for Mission Integration Technical Support activities in the US DoD Office of the Under Secretary of Defense for Research and Engineering. She leads the team supporting mission engineering activities for selected priority Defense missions and the application of digital engineering to mission engineering. She was the technical lead for development of the DoD guide for systems engineering applied to systems of systems (SoS) and was the project lead for International Standards Organization (ISO) 21839, the first ISO international standard on 'SoS Considerations for Systems Throughout their Life Cycle'. Prior to this, Dr. Dahmann was the Chief Scientist for the Defense Modeling and Simulation Office for the US Director of Defense Research and Engineering (1995-2000) where she led the development of the High-Level Architecture, a general-purpose distributed software architecture for simulations, now an IEEE Standard (IEEE 1516). Dr. Dahmann is a Fellow of the International Council on Systems Engineering (INCOSE) and the co-chair of the INCOSE Systems of Systems Working Group and co-chair of the National Defense Industry Association SE Division SoS SE Committee.

DAVID SÁNCHEZ GARCÍA

David Sánchez García holds a BS/MS in telecommunication engineering from UPM (Universidad Politécnica de Madrid). He has more than 30 years of experience at Isdefe in systems engineering with particular focus on electronic warfare and signal intelligence systems, where he is applying innovative systems of systems approaches in the Ministry of Defense.

Currently he is leading Isdefe's technical assistance to the Joint Signals Intelligence SANTIAGO Program Office of the National Armament Directorate (DGAM) SDG-PROGRAMAS.

He is certified as CSEP by INCOSE and as PMP by PMI. He is also involved as an in-house trainer on systems engineering at Isdefe.



CHAPTER 2

From Systems Engineering to Systems of Systems Engineering: An overview

Dr. Michael Yokell, *INCOSE (US Aerospace and Defense, retired) (Michael.Yokell@INCOSE.net)*
Dr. Alejandro Salado, *The University of Arizona (alejandrosalado@arizona.edu)*
Adolfo Sánchez Domínguez, *Isdefe (asdominguez@isdefe.es)*

Abstract

Engineering Systems of Systems (SoS) presents unique challenges that require a departure from traditional systems engineering (SE) practices. While most SE practices provide a structured framework for managing system life cycles, their direct application to SoS is limited by the degrees of governance, managerial independence, operational independence, and interoperability constraints among constituent systems (CS). Unlike traditional systems, where mandates and centralized control can ensure compliance, SoS must balance command, negotiation, and influence, depending on the autonomy of its components. This chapter provides a summary of the main adaptations that the application of systems engineering requires to be effective in the engineering of SoS.

Keywords

SoS engineering, operational independence, managerial independence.

1. INTRODUCTION

Applying traditional SE principles to SoS is far from straightforward. Unlike conventional systems, where engineers design and control every element, SoS presents a complex, decentralized, and evolving landscape, as described in the previous chapter. The core challenge is that an SoS is not built from the ground up. Instead, it emerges from the interaction of multiple, independently operated and managed systems, each with its own stakeholders, objectives, and operational constraints.

In this environment, the role of systems engineers transforms from designers and decision-makers to orchestrators and influencers, aligning diverse interests, fostering interoperability, and guiding system interactions to achieve a greater, emergent capability.

One of the most fundamental shifts in engineering SoS is recognizing that traditional SE approaches, which are highly prescriptive for SE processes, cannot be applied in a one-size-fits-all manner to SoS [1]. For a traditional system, most SE standards mandate what must be done: requirements must be managed, verification and validation must be performed, and all lifecycle processes are expected to be executed in a rigorous, disciplined manner. There is an implicit assumption that there is centralized control, meaning engineers can impose processes as needed to ensure system success.

However, when working with SoS, almost everything depends on the level of governance, managerial, and operational independence of its constituent systems (CS). Unlike a conventional system, engineers cannot assume mandates will be effective across the entire SoS. Some systems can be directly controlled, while others operate with complete independence, following their own priorities and decision-making structures. This means that, while in some cases mandates can still be used (especially for CS that are internally developed or contractually obligated to comply with the SoS structure), in other cases mandates are useless (particularly when dealing with autonomous systems that exist outside the SoS owner's jurisdiction). Most commonly, however, the solution lies somewhere in between, with degrees of negotiation, influence, and governance mechanisms used to align systems without imposing direct control.

This "it depends" is the essence of SoS engineering (SoSE). The degree of governance, managerial, and operational independence means that ownership of engineering decisions, whether they involve architecture, integration, verification, or sustainment, may be distributed because of the specific governance and managerial structures of the SoS. The key challenge of SoSE, therefore, is not just in applying engineering principles and methods, but in understanding where and how they can be applied given the autonomy, cooperation, and constraints of the CS involved.

Despite the increasing prevalence of SoS across industries, the methods and practices for engineering SoS are still in their infancy. While traditional SE has benefited from decades of refinement, with well-established standards, methodologies, and best practices [2], SoSE remains an emerging discipline, one where fundamental concepts are still evolving, and many challenges remain unresolved.

Traditional SE has been and continues to be formalized over time through applications in aerospace, defense, automotive, and industrial sectors. The processes outlined in standards like ISO/IEC/IEEE 15288 [3] and the INCOSE SE Handbook [4] provide clear guidance on how to approach system design, integration, verification, validation, and sustainment. This indicates that these methods are deeply ingrained in industry practices and are being actively refined through both theoretical research and real-world application.

In contrast, SoSE does not yet have the same level of maturity. Although foundational work has been done, particularly in defense and large-scale infrastructure, a large corpus of consolidated knowledge, concepts, methods, and practices is still accumulating. Unlike traditional systems, which have been abundant and with various classes of systems of the same kind, we not only have a very limited number of SoS, but they are inherently heterogeneous and dynamic. In fact, the challenges of governance, autonomy, and emergent behavior vary significantly across domains, making it difficult to establish clear guidelines (as stated earlier, "it depends"). In addition, while SE education is growing significantly, SoS are not yet central to the educational content. If any, they are treated as a side note or in a dedicated course, exemplifying the different depth to which traditional systems and SoS are addressed in these programs. The gap is amplified by the fact that engineering SoS requires a paradigm shift in engineering methods, as core assumptions for engineering practices are different: from direct control to coordination, from predictability to adaptation, and from closed design of individual components to enabling emergent capabilities. These changes require methods and approaches that have traditionally being foreign to engineering.

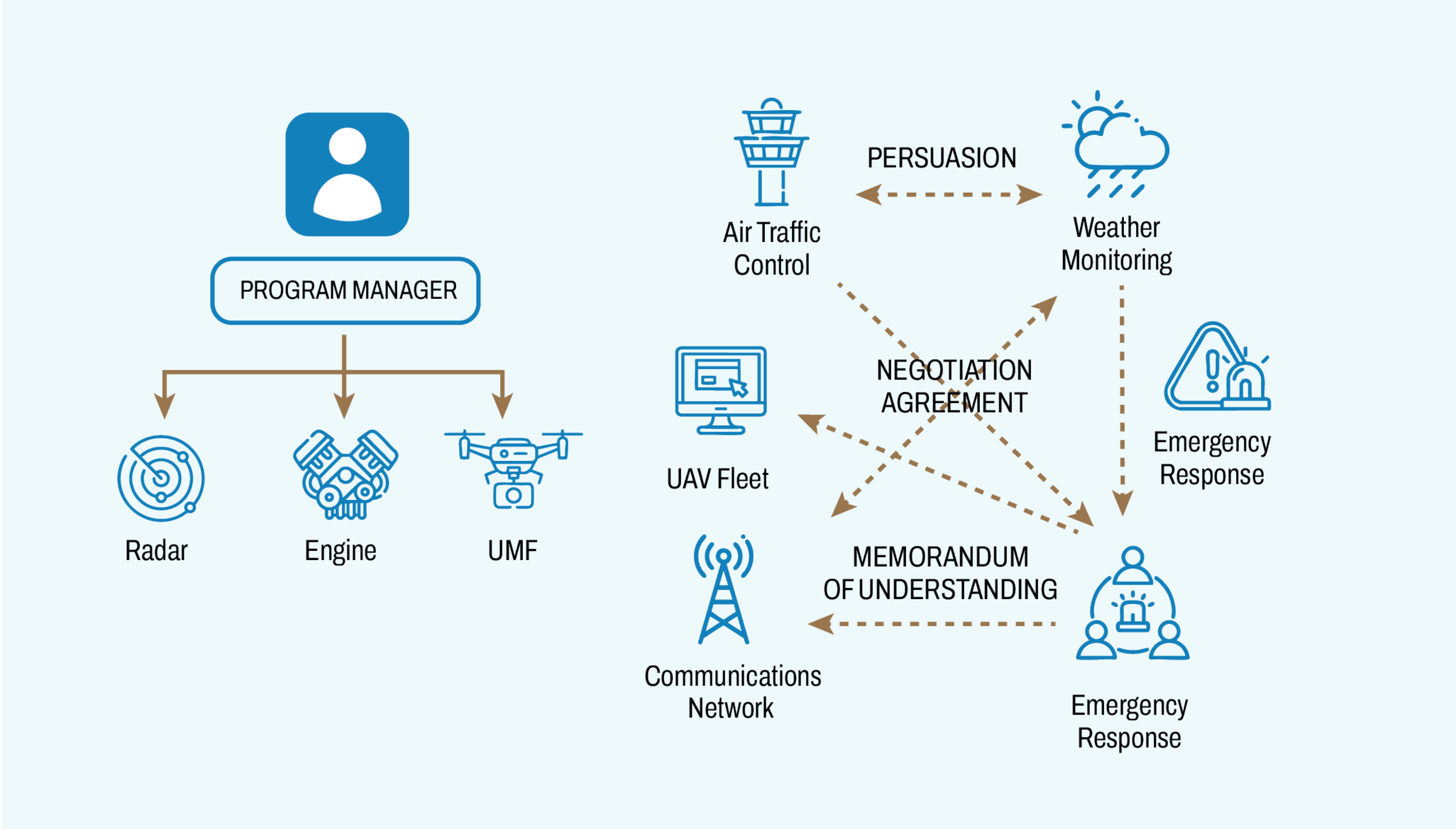


Figure 1. The degree of governance, managerial, and operational independence requires new system engineering approaches. (Left: traditional command and control; Right: SoS coordination).

Although SoSE is still in its early stages, progress is being made though. Research activity is growing, and several frameworks and methodologies are emerging. As of today, we can leverage guidelines, principles, and some isolated methods that are considered helpful to support SoSE. These have been compiled in three international standards:

- **ISO/IEC/IEEE 21839** [5] focuses on critical considerations for SoS during life cycle stages of constituent systems, aligned with ISO/IEC/IEEE 15288, and addresses the interaction and effective operation of constituent systems within an SoS.
- **ISO/IEC/IEEE 21840** [6] offers guidelines for applying ISO/IEC/IEEE 15288 in the context of SoS, exploring the similarities and differences between systems and SoS and detailing engineering practices for SoS.
- **ISO/IEC/IEEE 21841** [7] defines a taxonomy for SoS to improve understanding and communication among stakeholders, which also serves as a basis to characterize the SoS at hand to better guide the selection of SoSE practices.

We present an overview of such guiding principles in the next sections. Particularly, we present three key aspects of SoSE in Section 2 that provide an overview of some of the core shifts from traditional SE to SoSE, and then follow in Sections 3 through 6 with overviews of the guidance for applying traditional SE processes to SoS according to [6].

Note: Throughout the chapter, the term 'system' will be sometimes used to refer to traditional systems, different from SoS.

2. KEY CONSIDERATIONS OF SOSE

As described earlier, governance, managerial, and operational independence are the critical differences that demand a shift in SE practices for SoS. They give rise to interoperability, predictability, and evolution as key considerations to define SoSE practices.

2.1. Interoperability: A negotiated process, not a defined specification

Since SoS are rarely designed as a unified whole, one of the biggest challenges is making independently developed systems work together. Traditional SE assumes integration can be planned upfront, but in SoS, integration is often an ongoing and dynamic process.

A fully governed SoS can dictate strict interoperability requirements, forcing all CS to comply. However, in an SoS with limited governance, engineers must negotiate interoperability, sometimes convincing system owners to adopt standards or provide data-sharing agreements.

In this environment, traditional SE tools, such as interface control documents (ICDs), can still be useful, but they are not enough on their own. Engineers can also need to establish standards and common frameworks that systems voluntarily adopt, use incentives to encourage alignment (e.g., providing funding, technical support, or operational advantages), and design middleware and adaptive architectures that allow non-compliant systems to integrate without full standardization.

Therefore, instead of imposing interoperability as a requirement, SoS engineers often guide interoperability as a process of alignment and adaptation.

2.2. Emergent behavior and the limits of predictability

A defining feature of SoS is that capabilities and behaviors emerge from system interactions rather than being explicitly designed. In a traditional system, engineers carefully define what the system will do. In an SoS, capabilities often arise as a side effect of system collaboration, sometimes beneficial, sometimes problematic.

Because of this unpredictability and because many of their CS are operational in real-world environments, SoS cannot always be tested as a whole, challenging the adoption of traditional system verification and validation (V&V), where systems are expected to undergo a rigorous campaign through prototyping, simulation, and controlled testing. Instead, modeling, simulation, and real-world experimentation become essential tools to assess how well the SoS as a whole adapts to change, rather than just verifying whether each individual CS meets its requirements.

2.3. Engineering for continuous evolution

Unlike traditional systems, which often have clearly defined lifecycles, an SoS is always in flux. New systems join, old ones leave, and technological advancements constantly reshape the environment. The idea of a “finalized” SoS is a myth. Instead, engineers must design for continuous evolution.

This means that rigid, sequential development models do not work. SoSE requires incremental, adaptive processes that allow for change without disrupting existing capabilities. To manage this, engineers must think beyond just technical solutions. Political, legal, and contractual frameworks must also evolve alongside technical systems. Risk management, traditionally focused on technical failures, must now encompass organizational, economic, and geopolitical uncertainties as well.

3. AGREEMENT PROCESSES

3.1. Acquisition process

For a traditional system, the organization that owns the system is generally referred to as the acquirer, whereas the organizations that provide the system are referred to as suppliers. In the context of SoS, terms such as “consumer,” “participant,” or “partner” probably better capture the essence of their relationship. In the context of SoS, a consumer obtains the capabilities of CS, with or without explicit agreement, and without actually acquiring the CS that produced the capabilities. Unlike with systems, the suppliers continue to be managerially independent, and the CS remain operationally independent.

In addition to formal approaches like contracts, less formal approaches such as memoranda of agreement and memoranda of understanding can be effective in the management arrangements for some types of SoS. For some types of SoS, though, agreements can be informal, tacit, or absent. Accepting terms of use for a product or service is one type of agreement. Many SoS operate effectively even in the absence of agreements.

3.2. Supply process

Because CS remain managerially and operationally independent, CS already have acquirers for their product or service. Some CS can be willing to expand or reallocate resources to address the needs of an SoS acquirer, or agree to not change CS characteristics without notice, but some CS are not.

Agreements within SoS span a wide spectrum of formality, from contracts at one end of the spectrum, moving through less formal approaches such as memoranda of agreement, to memoranda of understanding to no agreements at all. For example, using the software can constitute an agreement to certain terms and conditions. In SoS where an explicit agreement does not exist, the supplier has no direct obligations. However, suppliers generally want their products and services to be used, so they have some motivation to help acquirers find usefulness in them.

4. ORGANIZATIONAL PROJECT-ENABLING PROCESSES

4.1. Life cycle model management process

CS have their own life cycles and life cycle models, and can be at different places within those models (ref. Figure 2). The life cycle model of an SoS can be different from those of the CS. Effective use of life cycle management process provides a context for CS joining and leaving the SoS. Organizations governing CS could have little insight or interest in the policies, procedures, and life cycle models of the SoS or another CS. However, responsibility, accountability, and authority arrangements with CS should be defined and understood.

Assessment of SoS life cycle models and processes for use by the organization should consider that SoS concerns, life cycle models, and processes could differ from those of the CS. The life cycle model and process of the SoS should recognize and accommodate the life cycle model and processes of the CS. For some SoS type, CS can be unable or unwilling to make such adaptations. However, adaptation to SoS concerns can include SoS roadmaps or broad strategies. SoS process, model, and procedure improvements should recognize that improvements needed to support the SoS are different from those needed to support the CS on its own and that the priorities of the SoS could differ from the priorities of the CS.

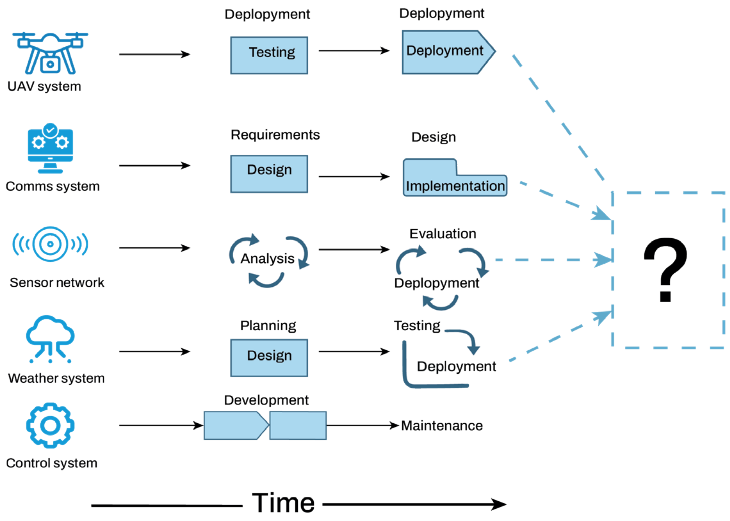


Figure 2. Each CS has its own life cycle model and can be at different stages.

4.2. Infrastructure management process

Infrastructure requirements for SoS projects can be different from infrastructure requirements for CS projects. For example, physical facilities such as integration labs and test labs can be needed by the SoS for interoperability testing to address SoS concerns. SoS can need project infrastructure elements that are not needed by any of the CS individually. Developing and acquiring SoS project infrastructure elements can be quite different from systems. Because SoS and CS can be at different stages within their life cycles, the availability, robustness, and resiliency of SoS project infrastructure can be of greater concern than with the CS alone.

4.3. Portfolio management process

Portfolio management is more complicated for organizations involved with SoS. For the SoS, the lack of control over the CS, which remain managerially and operationally independent, can be especially problematic. Likewise, CS participation in an SoS can be problematic due to consumption of resources beyond what the CS originally needed to meet its own stakeholder's needs. For some types of SoS, the influence to achieve the SoS capabilities can be created through highlighting the mutual benefits with the SoS and its stakeholders, as well as with the CS and their stakeholders. Disincentives should be addressed and minimized where possible.

Depending on the degree of operational and managerial independence, additional resources and budgets can be needed to support an SoS beyond what was needed for the CS. The accounting for these resources and budgets can be different from the CS because of their independence. SoS organizations should consider that other organizations participating in the SoS could choose to sustain or not sustain projects regardless of whether agreements and stakeholder requirements are being met.

4.4. Human resource management process

Human resources for an SoS address activities/processes that are distinct from the CS processes. Because SoSE differs from SE, skills required by SoS projects differ from other types of skills needed for CS. For example, because CS remain managerially independent, skills related to influence instead of direction can be especially important. Assessing the return on investment in an SoS project is immature, so care should be taken when resolving conflicts.

4.5. Quality management process

A quality management approach may be established for CS as well as the SoS. SoS depend on the CS for quality management of the CS. There can be variation in the management of quality by the CS, ranging from highly prescriptive across all CS to completely absent in some CS. Organizations participating in an SoS should adjust their policies, objectives, and procedures to accommodate these realities. SoS quality evaluation criteria should focus on the SoS capabilities which can be different from the CS capabilities. SoS organizations should consider how to align approaches to achieve the integrated SoS goals in the presence of variable quality systems.

4.6. Knowledge management process

Because CS are managerially independent, they have their own taxonomies and knowledge assets for their systems. It is possible that CS can collaborate with other CS or the SoS, but SoS should not assume that they will. SoS may use ISO/IEC/IEEE's knowledge management process, for example to define SoS-specific knowledge assets that can be needed. SoS knowledge assets should survive individual CS, especially when CS enter or leave an SoS. Because the CS reside in different organizations, the organizational knowledge, skills, and knowledge assets can be distributed across different organizations.

5. TECHNICAL MANAGEMENT PROCESSES

5.1. Project planning process

Organizations create projects to meet changing SoS objectives or capability shortfalls. Projects can create new SoS and SoS elements, leveraging existing CS to fulfil capabilities. Planning should recognize that the CS remain managerially and operationally independent from the SoS and the SoS organization. SoS planning should define the roles responsibilities and authorities of the SoS organization (if any) as well as CS organizations in the SoS plans.

5.2. Project assessment and control process

In the context of SoS, assessment is complicated, and control is sometimes impossible. Skills related to coordination, collaboration, and influencing can be especially important. For some SoS, it is not always possible to perform technical reviews of some of the CS. However, "proxy" reviews (i.e., without the CS owner present) can provide some insight.

Where possible, corrective action should be agreed upon between the governing and managing authorities of each CS and collaboratively decided and executed instead of directed as is common with traditional systems. The use of incentives or penalties can be considered for some SoS types. SoS plans should be adjusted based on the current and planned states of the CS.

5.3. Decision management process

As with systems, decisions can affect the system as a whole or just elements of the system. For SoS, decisions can affect the SoS as well as CS, so CS should be engaged in both the decision and analysis processes along with the SoS. This is not always possible because some CS are not aware of their participation in an SoS. Managerial independence of the CS means that the CS manage their own decisions, some of which can be contrary to SoS interests.

5.4. Risk management process

Risks faced by the SoS differ from those faced by the CS. Risks to the SoS can result from CS that change, join, or leave the SoS or do not operate consistently with SoS expectations. Analysis of SoS risks can require analysis of the network of interoperating CS as well as candidate replacement CS if CS change or leave unexpectedly. Given that CS remain operationally and managerially independent, CS incentives can be considered to facilitate proper treatment of SoS risks. Enforcing risk treatment by CS is not always feasible.

5.5. Configuration management process

SoS configuration management focuses on system elements that specifically relate to the SoS. In contrast, CS retain configuration management of their systems. Because SoS functions are allocated to the CS, interfaces between each CS should be well defined, especially when the SoS does not control them. In the context of SoS, “controlled” means monitored and recorded, not that there is control exercised into approving/rejecting/implementing the change itself.

Baselines for the SoS depend on what, if any, agreements are in place. While cooperation from CS is helpful, it is not always possible, especially in loosely organized SoS. It can be possible to baseline or define compatibility across elements to map system features. Changes to items under configuration management can be controlled or uncontrolled from the perspective of another CS or the SoS.

Some aspects and attributes of the CS can be related to certification of interoperability. These attributes themselves can be leveraged to establish a baseline from the SoS point of view. Interface specifications can form a part of a baseline. Compatibility should be considered when the specifications change. Depending on the governance independence of the SoS, formal releases and deliveries of SoS capabilities can be infeasible. Instead, they can occur when CS capabilities are released and delivered.

5.6. Information management process

The types of information to be managed can relate to the SoS itself, the governance of the SoS and the CS. SoS information management should be clearly defined and agreements made with the CS, where possible, to share the needed information. Note that for some SoS types, CS can be unaware or unwilling to share information. SoS (and other CS) should respect CS confidentiality, security, and ownership of intellectual property, especially when agreements are informal or absent. Some information from CS is not always available to SoS stakeholders or other CS stakeholders.

5.7. Measurement process

Informational needs of both the SoS and the CS necessary for CS participation in the SoS should be identified. The definition of these measures should be accompanied by a description of the impact on the performance or capability of each CS, if applicable, and of the overall SoS. For some SoS types, collection, verification, and storage of data is not always possible. Where available, alternative approaches should be defined. Additional SoS elements can also be added to assist the monitoring of the SoS. Depending on the governance independence of the SoS, information provided by CS can be subjective, erroneous or absent.

5.8. Quality assurance process

Ideally, criteria and methods for quality assurance evaluations should be agreed upon. As with systems and subsystems, the agreement should identify roles and responsibilities as well as information, data, and assumptions in relation to the planned evaluations. In some cases, quality assurance activities can be handled by the SoS, or separately by each CS, without agreement between them. For some SoS type, CS can be unaware of their participation. If possible, problem treatment and definition of priorities should be agreed upon between the governing and managing authorities of each CS and the SoS.

6. TECHNICAL PROCESSES

6.1. Business or mission analysis process

Because the problems and opportunities addressed by SoS are typically broader and more complex than those addressed by systems, business or mission analysis can require the use and integration of modeling techniques and tools that are more heterogeneous than with systems. Ideally, the characterization of a solution space for an SoS should include the candidate CS, how each CS supports the new problem or opportunity, and any constraints these CS can impose on the solution space.

Since CS that are already deployed when the SoS is envisioned have existing operational considerations and constraints, the SoS can be affected and constrained. System approaches that assume the ability to change the CS likely will not be effective. Depending on the degree of managerial independence, different approaches can be necessary. For SoS, multiple alternative solutions can be viable.

6.2. Stakeholder needs and requirements definition process

Stakeholder needs and requirements definition process for SoS has the same focus as with systems, but identification and access to stakeholders can be constrained. It is not always possible to identify all stakeholders; elicitation of stakeholder needs will likely be incomplete and perhaps incorrect. The SoS should plan to accommodate emergent stakeholders and their needs as well as any constraints that potentially affect SoS operation and evolution. For SoS, CS continue to have managerial and operational independence as well as interdependence. It is important to recognize where key CS stakeholder needs align or conflict with the SoS objectives. To be successful, SoS should avoid infringing on CS objectives.

CS have their own operational concepts and lifecycles that evolve independently, so it is important to document where each system is within its lifecycle. There can be different periods of stability for some CS and instability for other CS. CS constraints on the SoS should be identified.

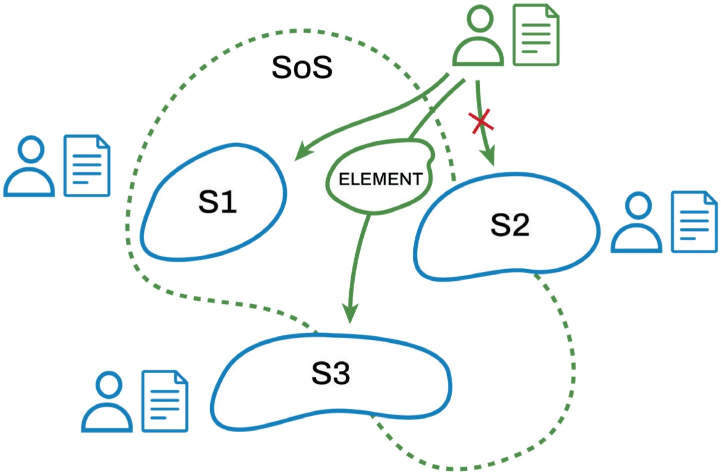


Figure 3. CS have their own stakeholders with their own requirements.

SoS and CS stakeholders should strive for agreement where possible, but agreement is not necessarily required of all CS stakeholders for effective definition of the SoS. However, care should be taken because disagreement can cause CS to exit the SoS or impede SoS objectives.

6.3. System requirements definition process

As with systems, the SoS requirements definition process transforms the stakeholders' desired outcomes into SoS requirements, characteristics, attributes, functions, and performance that the SoS should possess to satisfy the stakeholder requirements. For some types of SoS, CS owners can conduct or support SoS requirements definition. For other types of SoS, the CS owner can be unwilling to disclose this information. As with systems, SoS requirements and design constraints can change over time. As CS change, join or leave the SoS, the set of available capabilities changes, affecting what is feasible for the SoS to provide. Requirements that were previously met can be unmet without warning.

Because CS remain managerially and operationally independent, CS do not always accept and implement SoS requirements. Consequently, SoS sometimes do meet their objectives. However, requirements definition should consider critical performance measures for the CS and SoS to facilitate delivery of SoS capabilities under various situations. As with systems, SoS stakeholder requirements can trace to SoS capability objectives and CS or system element requirements. Unfortunately, SoS requirements can conflict with existing CS attributes.

6.4. System architecture definition process

As with systems, the architecture definition of SoS focuses on the critical functionality of the elements. With SoS, these include CS, system elements, and their interactions. Insight into the inside architecture of CS can be unnecessary or even impossible. Systems are often decomposed in terms of hierarchies; SoS are often described as networks with complex interconnections. Consequently, SoS architecture can be limited based on partial information from the CS, some of which can be unaware of their participation in the SoS.

The SoS architecture should expect a need to define system elements that are not part of any CS. Likewise, CS change, join and leave the SoS over time, so the architecture definition can remain fluid. As with systems, SoS architectures should reflect the needs of the SoS stakeholders for the SoS capabilities. However, it is also important to acknowledge in the architectures the needs of the CS stakeholders which can be affected by CS participation in the SoS. Ideally, SoS architectures should address SoS while minimizing or avoiding adversely affecting the CS.

6.5. Design definition process

As with systems, an overarching design for the SoS is needed. However, unlike systems, the system elements of an SoS can be CS that are owned by other organizations and operated for their own purposes. However, system approaches as in ISO/IEC/IEEE 15288 can be applied to specific system elements that are not part of any CS. Multiple design definitions can be necessary to implement changes into the SoS on a time-phased basis, especially when CS join or leave the SoS unexpectedly.

SoS Design Definition should acknowledge that desired changes to a CS are not possible and that the CS design and capabilities must be accepted and accommodated as is. To the extent possible, SoS Design Definition should consider time phasing of the CS implementations so that CS capabilities remain compatible with each other at all points in time.

For SoS, design definition emphasizes the selection and adaptation of CS or other system elements that can be necessary to facilitate interaction of the CS in the SoS. Alternative CS should be assessed for viability, not just technically, but the implications of their operational and managerial independence. A willing participant can be advantageous of a technical superior, but unwilling participant.

The selection of the set of CS to meet SoS needs is allocates systems, not just system elements to SoS requirements. The SoS design definition process focuses on assessing the ability of CS and system elements, either current or proposed, to implement the interfaces to meet SoS needs.

Organizations managing CS can be unwilling to fully disclose design information. Fortunately, SoS do not need to understand the internal aspects of CS designs, but can be more narrowly concerned with the interfaces and external characteristics of the CS. That is, in SoS design definition, it is important to have just enough CS design information to understand their behavior, but without access to the details of how CS achieve it.

6.6. System analysis process

Like systems, SoS system analysis can leverage actual and predicted data. Because many CS pre-exist the SoS, actual data can be available from their operation, though CS owners can be unwilling to share this information. For this reason, predictive data separate from the CS can be necessary.

It is possible that the System Analysis process can be performed by each CS owner to support CS-level decision management with SoS impacts. However, CS owners have different viewpoints and interests from the SoS, so their analyses and results can differ from those of the SoS. Unfortunately, CS decisions based on their own analyses can conflict with SoS interests and requirements. Resolution of conflicts can be very challenging or impossible.

SoS analyses should explore and validate assumptions and results related to operational and managerial independence of the CS. SoS should consider adding measuring points and instrumentation to provide the data to validate analyses or models.

6.7. Implementation process

SoS are typically implemented by composing existing and potentially modified CS along with other system elements to provide new capabilities. Constraints from existing CS influence the requirements, architecture, or design. Like systems, the remaining system elements need to be realized and the implementation process from ISO/IEC/IEEE 15288 can be used.

SoS information management and support infrastructure can enable the SoS to flourish and develop as needed. Time phasing of CS changes should be carefully planned. Ideally, each CS should be expected to continue to support its own independent capabilities and any changes to the CS are ready to support the desired SoS capability. To the extent possible, care should be taken to confirm that desired interactions and capabilities are not damaged by CS changes.

6.8. Integration process

SoS often consist of existing CS that are integrated to meet the requirements of the SoS. Constraints on integration that influence SoS requirements, architecture, or design, including interfaces, can be extensive. Unlike systems, integration for the SoS is often performed with operational CS in an operational SoS environment. SoS Integration usually occurs within the context of an evolving, continuously operating SoS. It can be impossible or impractical to halt operation. Consequently, planning for integration should facilitate interaction with ongoing CS and SoS operations.

Because CS can change, join or leave an SoS, frequent integration can be expected. Frequent checking of the interfaces between the CS and implemented system elements can be important.

6.9. Verification process

Verification of SoS requirements is not usually possible to a similar degree of certainty and fidelity as is possible with other types of systems. However, verification activities can still be valuable to assess how well the SoS is meeting the requirements to the extent that they are known and understood. Unlike many systems, SoS verification almost always occurs in the context of operations that are ongoing and changing. Verification should be planned and executed with ongoing operations in mind.

For some SoS, CS owners can perform some verification of their partial views of the SoS. For others, it can be impossible to test some SoS capabilities due to safety, security, or cost. SoS Verification can rely on modelling or analysis. Unlike systems, verification of some CS is not guaranteed. Complete verification of the SoS and resolution of anomalies can be infeasible.

6.10. Transition process

As with systems, the transition process establishes the desired capabilities in an operational environment. For some CS, new features and capabilities can be available to stakeholders frequently, while other CS make transitions at a much slower cadence. Consequently, transition events by CS can occur just as frequently for the SoS. Some SoS owners can influence the phasing of CS changes, while other CS can be unwilling to adapt to the needs of the SoS. To the extent possible, transition events should be planned and executed to coordinate CS and SoS capabilities.

CS are usually in operation prior to the SoS capability being transitioned to operations. It is possible that CS can join and leave an SoS unexpectedly. Consequently, delivery of the SoS capabilities is not guaranteed. SoS governance can consider incentivizing CS to be created in certain areas.

6.11. Validation process

Validation of the SoS validation can occur as an ongoing process throughout operations, rather than a singular event as typically happens with systems. Access to objective evidence about some SoS objectives or stakeholder requirements can be limited or even absent due to safety, security, or operational considerations. In such cases, planning for SoS validation should consider the availability of subjective or ancillary evidence.

As with the validation of system, validation of SoS can identify discrepancies between the original requirements and the operational capabilities that can be provided. SoS stakeholders as well as their CS stakeholders, can have conflicting goals. Thus, unlike systems, it can be impossible to obtain full stakeholder ratification of the SoS validation. For some SoS, CS owners can validate a partial view of the SoS that is available to each CS owner, while others can be unwilling to support SoS validation. For many SoS, service availability of the SoS is not guaranteed.

Interestingly, validation of CS can be irrelevant or unnecessary if the stakeholder validates the SoS. However, non-validated, strongly independent CS can reject or abandon the SoS, posing a risk to the availability of SoS services and capabilities.

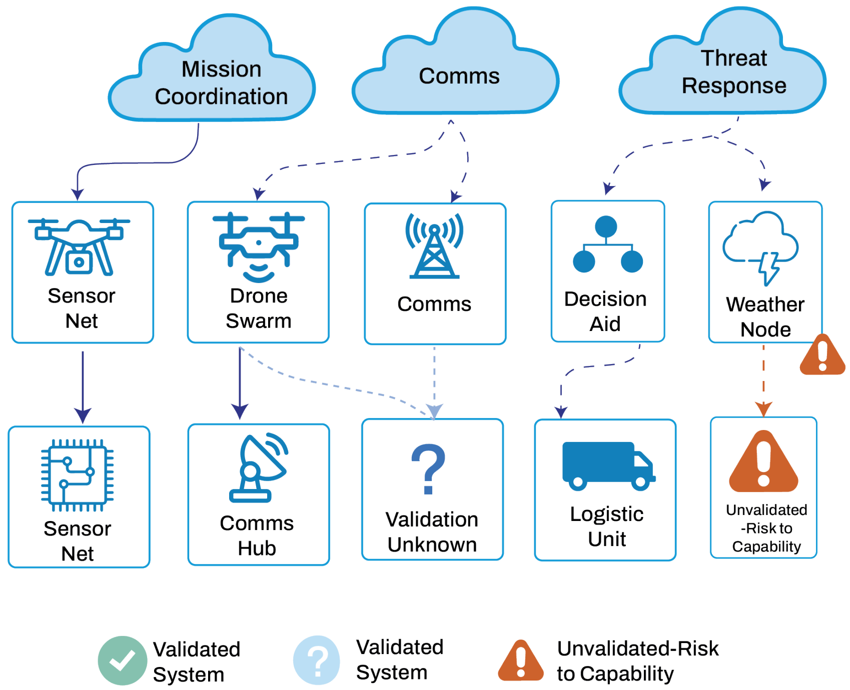


Figure 4. Some CS can validate partial SoS views, but non-validated CS pose risks to service availability.

6.12. Operation process

SoS are often assembled or orchestrated from exiting CS that have their own operational priorities and constraints. Sometimes it is not possible to feed these priorities and constraints to the other processes in a timely manner. Operational changes to the CS and their interfaces affect the SoS. Likewise, stakeholder feedback on the operation of the SoS can suggest that changes should be made to the CS, but the CS can be unwilling to make them because they have their own operational interests that can differ from those of the SoS.

6.13. Maintenance process

Maintaining the SoS is particularly challenging because organizations that own and manage CS can make changes to them for their own purposes, potentially affecting the SoS and the interfaces to other CS. These changes can occur without prior warning or coordination within the SoS. For some types of SoS, organizations managing CS can be unwilling to report pending changes to address corrective, perfective, or adaptive maintenance. Likewise, these organizations can be unwilling to share data related to failures. Planning for the SoS can be constrained by the lack of access to data that would typically be available within a system. Consequently, the ability for an SoS to maintain itself can require proactive contingency planning.

6.14. Disposal process

Because CS within the SoS are independently governed and managed, it is possible, and perhaps likely, that CS can exit the SoS without much coordination or planning. An organization can decide to dispose of a CS, adversely affecting any SoS it was participating in. For some type of SoS, the CS can be unaware of the participation. Conversely, an SoS can be retired and the outcomes of the disposal process achieved without adversely affecting the CS, which can continue to be operated for their own purposes. Unlike systems that are not SoS, the SoS can simply disconnect from the CS rather than disposing of the CS. However, if the SoS has enabling systems, services, or system elements unique to the SoS that are not needed by any of the CS, the disposal process in ISO/IEC/IEEE 15288, clause 6.4.14 may be followed and the outcomes of the disposal process achieved.

7. CONCLUSIONS

Applying systems engineering to SoS is not just about adapting existing processes, it is about embracing a fundamentally different way of thinking. Instead of focusing on control, predictability, and optimization, SoS engineering is about influence, adaptation, and resilience. This requires a shift in several areas:

1) The role of systems engineers must evolve from designer to orchestrator.

2) Influence must replace direct control as the primary means of ensuring system alignment.

3) Interoperability must become a fundamental requirement, not an afterthought.

4) Emergent behavior is anticipated and managed rather than avoided, and

5) V&V shifts from static, requirement-driven testing to continuous evaluation in real-world contexts. However, it is also important to note that the development of dedicated approaches and methods to support this shift are in their infancy.

Change is necessary but it should be pursued with caution[1].

[1] *Caution does not mean sticking to traditional systems engineering practices. Caution here means that extra care must be taken when dealing with the engineering of SoS; an SoS is not simply a larger system…*

REFERENCES

1. McDermott, T. and V. Ramos, New Kinds of Systems, in Introduction to Systems Engineering in the 21st Century, A. Salado, Editor. 2024, Isdefe: Madrid, Spain. p. 35-55.

2. Salado, A., A. Sancehz, and D. Verma, Systems Engineering in the 21st Century, in Introduction to Systems Engineering in the 21st Century, A. Salado, Editor. 2024, Isdefe S.A.: Madird, Spain. p. 15-33.

3. ISO, ISO/IEC/IEEE International Standard - Systems and software engineering -- System life cycle processes. ISO/IEC/IEEE 15288 First edition 2015-05-15, 2015: p. 1-118.

4. INCOSE, Systems Engineering Handbook. A Guide for System Life Cycle Processes and Activities. 5th ed. 2023, Hoboken, NJ, USA: John Wiley and Sons, Inc.

5. ISO/IEEE/IEC, ISO/IEC/IEEE 21839, Systems and software engineering — System of systems (SoS) considerations in life cycle stages of a system. 2019.

6. ISO/IEEE/IEC, ISO/IEC/IEEE 21840, Systems and software engineering — Guidelines for the utilization of ISO/IEC/IEEE 15288 in the context of system of systems (SoS). 2019.

7. ISO/IEEE/IEC, ISO/IEC/IEEE 21841, Systems and software engineering — Taxonomy of systems of systems. 2019.

APPENDIX: COMPARISON BETWEEN SYSTEMS ENGINEERING PRACTICES FOR SYSTEMS AND SOS

	Processes	System	SoS
Agreement Processes	Acquisition	A single acquirer manages the system and contracts suppliers for its development and operation, with formal agreements.	There may be no single acquirer; CS may remain operationally and managerially independent, with agreements that may be formal, informal, or even nonexistent.
	Supply	The acquirer has full managerial and operational control over the system and its suppliers.	CS may remain independent with their own acquirers. Adaptation to SoS needs varies, and agreements may range from formal contracts to none. Suppliers may have no direct obligations with SoS unless explicitly agreed.
Organizational Project-enabling Processes	Life cycle model management	Follows a well-defined life cycle model managed by the acquirer. When different life cycle models are used in different parts of the system, they are still integrated at the system level.	Each CS may have its own life cycle model, which may be at different stages, and are not necessarily integrated towards an aggregated life cycle model.
	Infrastructure management	Infrastructure requirements are defined for a single system and its needs.	May require additional infrastructure, such as integration and test labs, to ensure interoperability. Availability and resiliency are critical due to potentially varying CS life cycle stages.
	Portfolio management	Portfolio management is simpler, as the organization has full control over its system and resources.	Portfolio management may be more complex due to the lack of control over independent CS, which can require additional resources and budgets.
	Human resource management	Human resources focus on activities and processes specific to the system, with skills tailored to managing the system itself.	Human resources in SoS may require different skills, particularly related to influence rather than direct control, due to the managerial independence of CS.
	Quality management	A quality management approach is established and managed for the system itself.	SoS depends on the quality management of the CS, which may vary significantly. SoS quality evaluation focuses on integrated capabilities rather than individual CS capabilities.
	Knowledge management	Taxonomies and knowledge assets are defined for the system itself.	SoS defines its own knowledge assets, which must endure across CS transitions. Collaboration between CS or with the SoS is not guaranteed, and knowledge is distributed across different organizations.

Technical Management Processes	**Project planning**	Projects are created to fulfill specific system objectives and are fully managed within the system's control.	Projects address capability shortfalls by leveraging existing CS. Planning must account for the independence of CS and define roles, responsibilities, and authorities of both SoS and CS organizations.
	Project assessment and control	Assessment and control are straightforward and managed directly within the system.	Assessment is complex with limited control. Success relies on coordination, influence, and collaboration. Corrective actions are collaborative, and plans should adapt to the evolving status of CS.
	Decision management	Decisions impact the system as a whole or specific elements, with full control over the system's decisions.	Decisions impact both the SoS and CS. CS may not be always aware of their role, and their independent decisions can conflict with SoS goals.
	Risk management	Risks are contained within the system and are managed directly.	Risks arise from changes, departures, or inconsistencies in CS. Risk analysis involves understanding the network of interoperating CS and potential replacements. Enforcing risk treatment is not always feasible.
	Configuration management	Configuration management is handled internally within the system, focusing on its elements.	Configuration management focuses on elements related to the SoS, with CS retaining control over their own systems. Interface specifications and compatibility are critical, and changes can be uncontrolled from the SoS perspective.
	Information management	Information management is focused on the system and its internal governance.	Information management involves both the SoS and CS, potentially requiring clear agreements for sharing. CS may be unwilling or unaware of the need to share information, and confidentiality, security, and intellectual property must be respected. Some information may not be readily accessible to all stakeholders.
	Measurement	Information needs are clearly defined within the system for its own performance.	SoS needs to define and manage information across both the SoS and CS, but data can be incomplete, subjective, or unavailable.
	Quality Assurance	Quality assurance criteria and methods are clearly defined and agreed upon.	Quality assurance can be handled by the SoS or independently by each CS, sometimes without formal agreements. CS can be unaware of their participation, and priorities and problem treatments need to be agreed upon between CS and SoS authorities.

Technical Processes	**Business or mission analysis**	Problems are typically simpler, and analysis uses more homogeneous modeling tools.	Problems are broader and more complex, requiring diverse modeling tools and integration. CS constraints and operational considerations affect the SoS, and multiple alternative solutions can be viable.
	Stakeholder needs and requirements definition	Stakeholder needs and requirements are identified and addressed with a clear focus.	Identifying and accessing all SoS stakeholders can be constrained, and needs can be incomplete or incorrect. Emergent stakeholders and conflicts with CS objectives must be managed.
	System requirements definition	Requirements are defined clearly to meet stakeholder outcomes, with stable characteristics, functions, and performance.	Requirements are subject to change as CS join, leave, or change, affecting feasibility. CS are not obliged to accept or implement SoS requirements, and conflicts between SoS and CS requirements can occur.
	System architecture definition	System architecture is often hierarchical, focused on critical functionality and internal components.	Architecture is network-based with complex interconnections between CS and system elements, and can be limited by partial information from CS.
	Design definition	Systems require a comprehensive design that integrates all components and elements within a defined scope.	Design must account for system elements (CS) owned and operated by different organizations for independent purposes. Design should consider changes on a time-phased basis, acknowledging that CS design and capabilities cannot be altered. SoS design focuses on selecting and adapting CS or system elements for compatibility and functionality.
	System analysis	System analysis uses both actual and predictive data to assess and optimize system performance.	SoS analysis uses actual and predictive data, but CS can be unwilling to share real data. SoS analysis should account for the operational and managerial independence of CS, adding measurement points to validate assumptions and results.
	Implementation	Systems are typically implemented by developing and integrating new system elements based on requirements and designs.	Implemented by composing existing CS, potentially modified, along with other system elements to create new capabilities. Constraints from existing CS impact the SoS requirements, architecture, and design.
	Integration	Integration is typically performed in a controlled environment, where new or modified system elements are integrated based on a defined architecture and design.	Integration involves existing CS integrated to meet SoS requirements, often within an operational environment. Integration occurs within a continuously operating SoS, and it can be impractical to halt operations.
	Verification	Verification is typically performed in a controlled environment with a high degree of certainty, confirming that the system meets its requirements.	Verification is more challenging and less certain, often occurring in an evolving operational environment. It can be impossible to fully verify some capabilities due to safety, security, or cost constraints. Verification is often reliant on modeling, analysis, and partial testing of CS.

	Transition	Transition involves moving a system from development to an operational environment, with a clear focus on ensuring it meets desired capabilities.	Transition in SoS involves coordinating multiple CS, which can have different transition cadences. SoS governance can incentivize CS to align with SoS goals, but transitions are not guaranteed. Frequent planning and coordination are required.
	Validation	Validation is typically a one-time event, ensuring the system meets its intended requirements and objectives. It involves gathering objective evidence to confirm the system's capabilities.	Validation is an ongoing process, influenced by the operational context. Access to objective evidence can be restricted, and subjective or ancillary evidence can be used. Discrepancies between requirements and operational capabilities can arise. Full stakeholder ratification can be impossible due to conflicting goals between SoS and CS stakeholders.
	Operation	Operational changes typically follow a planned, centralized process, affecting the system as a whole.	CS have their own operational priorities and constraints. Operational changes to CS and their interfaces impact the SoS. CS can resist changes suggested by SoS due to differing operational interests.
	Maintenance	Maintenance is typically planned and controlled within a closed environment, with direct access to data.	Maintenance is challenging due to independent CS, who can make changes without coordination or notification, potentially affecting the SoS. Proactive contingency planning is necessary.
	Disposal	Disposal is managed within a controlled environment, and systems are typically retired or decommissioned according to established procedures.	CS can exit the SoS without coordination, and the SoS can simply disconnect from the CS without disposing of it.

DR. MIKE YOKELL

Dr. Mike Yokell is a retired systems engineering leader with more than 40 years in the US aerospace and defense industry. He is the editor, co-editor or contributor to numerous international standards on systems and software engineering. Mike is certified as an expert systems engineering professional by INCOSE. He holds multiple US and European patents for model-based systems engineering and large-scale immersive virtual reality. He contributes regularly to the INCOSE Agile and System of Systems working groups.

DR. ALEJANDRO SALADO

Dr. Alejandro Salado is an associate professor of systems engineering with the Department of Systems and Industrial Engineering at the University of Arizona and the director of the systems engineering program. In addition, he provides part-time consulting in areas related to enterprise transformation, cultural change of technical teams, systems engineering, and engineering strategy. Alejandro conducts research in problem formulation, design of verification and validation strategies, model-based systems engineering, and engineering education. Before joining academia, he held positions as systems engineer, chief architect, and chief systems engineer in manned and unmanned space systems of up to $1B in development cost. He has published over 150 technical papers, and his research has received federal funding from the National Science Foundation (NSF), the Naval Surface Warfare Command (NSWC), the Naval Air System Command (NAVAIR), and the Office of Naval Research (ONR), among others. He is a recipient of the NSF CAREER Award and the International Fulbright Science and Technology Award. Dr. Salado holds a BS/MS in electrical and computer engineering from the Polytechnic University of Valencia, a MS in project management and a MS in electronics engineering from the Polytechnic University of Catalonia, the SpaceTech MEng in space systems engineering from the Technical University of Delft, and a PhD in systems engineering from the Stevens Institute of Technology.

ADOLFO SÁNCHEZ DOMÍNGUEZ

Adolfo Sánchez holds a degree in Computer Engineering from the University of Zaragoza and a degree in History from UNED. He has been involved in defining and developing a modular avionics architecture evaluator, maintaining Eurofighter software, developing quality assurance software for electronic warfare strategic programs, and acquiring, deploying, and implementing the CIS Program for the Military Emergencies Unit. Currently, he serves as the CIS Defense Area Coordinator at Isdefe, supporting the Ministry of Defense's Digital Transformation Subdirectorate. He is a Certified Systems Engineering Professional (CSEP) by INCOSE and internal trainer for various systems engineering courses at Isdefe

GRAVITY FIELD MAP
GLOBAL TEMPERATURE
SEA LEVEL
ALTITUDE
INFO
Information about USA
Government Database
Global Position Localization
Warning Sistem
Fire alarm system
Protocol
TRACKING PROGRESS
File.log.
Debug
Satelist Results
INTEGRATED DATA FEEDS
ZOOM LEVEL: 4X
CAMERA: IR4T
SATELITE LINK
ACTIVE
SAFETY PROTOCOLS

CHAPTER 3

Mission Engineering

Dr. James Moreland Jr, *Retired US DoD Senior Executive and Raytheon Technologies Vice President (james.moreland.mei@gmail.com)*
LtCol. Víctor M. Sobrino García, *Spanish Ministry of Defense (victor@sobrinosastre.com)*

Abstract

Mission Engineering (ME) is emerging as a structured, mission-centric methodology designed to support the development and integration of complex Systemsm of Systems (SoS) across operational domains considering the mission and the operational expertise as the focus of the discipline. ME provides a repeatable and data-driven framework that links engineering rigor to strategic and operational insights, enabling capability development that aligns with defined mission outcomes. Using Mission Engineering Threads (METs), Effect Webs, and structured frameworks such as DOTMLPF-P or MIRADO-I, ME enables decision-makers to map, assess, and evolve the interactions between systems, people, and processes critical to mission success. The ME methodology integrates risk-based assessments, formal analysis of integration and interoperability, and the generation of mission success criteria—from strategic intent to tactical performance—creating a common reference architecture that ensures alignment across stakeholders, domains, and coalition partners. ME fosters a shift from platform-centric optimized acquisition to capability-focused development, enabling resilient, interoperable, and future-ready mission architectures that support both defense and broader industrial objectives.

Keywords

Mission Engineering, Systems-of-Systems, Mission Engineering Thread, Capability Development, Operational Effectiveness, Integration, Interoperability, DOTMLPF-P , MIRADO-I, Effect Webs, Resiliency.

1. INTRODUCTION

Mission Engineering (ME) is a mission-centric approach for Systems of Systems (SoS) that require a better understanding of the interactions and dependencies between the multitude of systems necessary to execute complex, multi-domain operational missions. Originally, ME was developed and implemented in the United States (US) Department of Defense (DoD) as a methodology to pursue the effectiveness of collaborating SoS based on overarching mission objectives as key to define mission success. In essence, ME provides a systematic, quantifiable, and repeatable approach that supports strategic decision making across ecosystems, supporting a successful transition of platforms and systems into SoS capabilities.

The ME discipline links engineering rigor to operational and business insights necessary to identify capability-related requirements and solutions in alignment with the mission of the overarching organization. Formally, ME has been defined as "an interdisciplinary process encompassing the entire technical effort to analyze, design, and integrate current and emerging operational needs and capabilities to achieve desired mission outcomes" [1]. In practice, mission engineers plan, analyze, organize, and integrate operational concepts for the purpose of evolving the end-to-end operational architecture and capability attributes. In a defense context, this takes place across the Doctrine, Organization, Training, Materiel, Leadership and Education, Personnel, Facilities, and Policy (DOTMLPF-P) spectrum (in Spain MIRADO-I, Material, Infraestructura, Recursos Humanos, Adiestramiento, Doctrina, Organización, e Integration), including adversary and competitor behaviors. Despite its specificities regarding the application of military force to success in the operational arena, ME provides a set of tools and methods to enable organizations across industries to take advantage of its benefits.

ME analysis integrates authoritative data and a common framework to produce SoS architectures and highlighted capability attributes that inform requirements and establish data driven technical architecture baselines. The ME outputs inform stakeholders and decision makers across the industrial sector ecosystem, as captured in Figure 1 for the US DoD, which shows the methodology from defining the problem statement or operational need to the development of potential solutions.

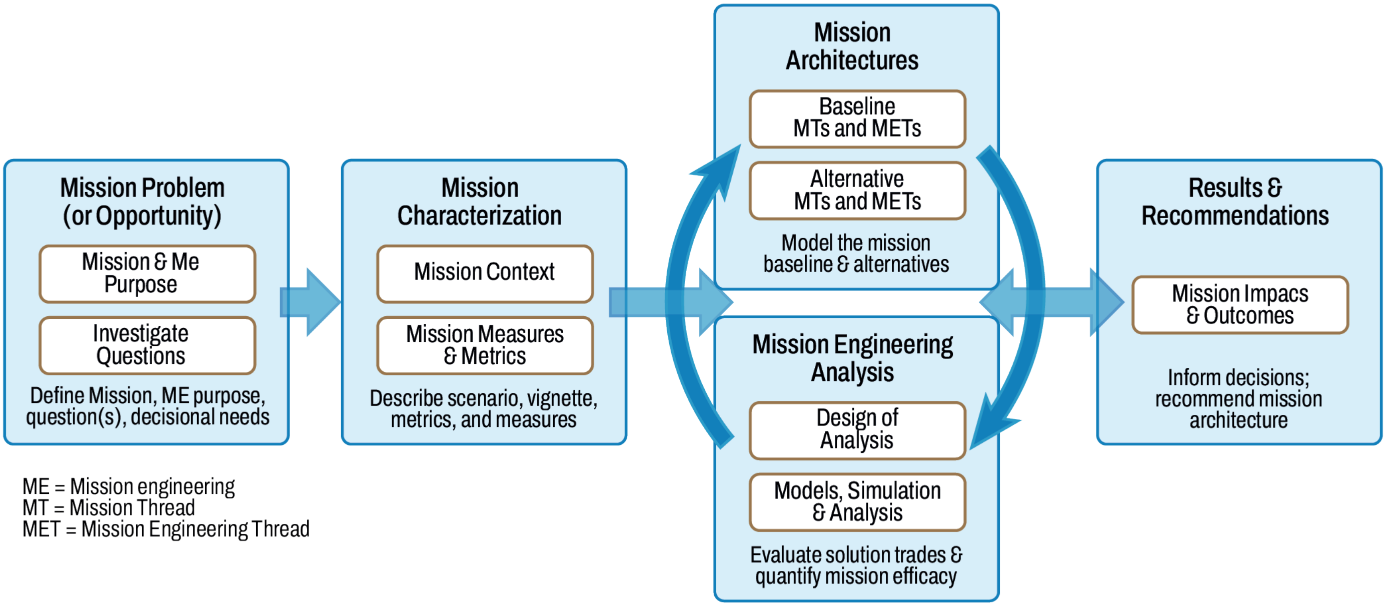

Mission Problem (or Opportunity)
Mission & Me Purpose
Investigate Questions
Define Mission, ME purpose, question(s), decisional needs
Mission Characterization
Mission Context
Mission Measures & Metrics
Describe scenario, vignette, metrics, and measures
Mission Architectures
Baseline MTs and METs
Alternative MTs and METs
Model the mission baseline & alternatives
Mission Engineering Analysis
Design of Analysis
Models, Simulation & Analysis
Evaluate solution trades & quantify mission efficacy
Results & Recommendations
Mission Impacts & Outcomes
Inform decisions; recommend mission architecture
ME = Mission engineering
MT = Mission Thread
MET = Mission Engineering Thread

Figure 1. OSD R&E ME methodology.

Institutions such as the Naval Surface Warfare Center Dahlgren Division (NSWCDD) have pioneered ME implementation within the US. While Europe does not currently have an exact counterpart, initiatives within the European Defence Agency (EDA) or national organizations (e.g., Spain's DGAM) are beginning to explore similar approaches.

Once the mission success is understood with identified mission success criteria, the operational capability and specific mission can be determined for execution. An operational capability can be described from both structural and behavioral perspectives. The first is focused on how a capability is structured (DOTMLPF-P for USA, DOTMLPF-I for EU, MIRADO-I for Spain); the second defines what outcomes the capability provides, and therefore, defines what operational activities (OA) are fulfilled by this capability.

This combined SoS behavioral approach and structured implementation has demonstrated tremendous success in increasing the likelihood of integration and interoperability with respect to the total SoS effectiveness, while moving away from the optimization of individual systems following a platform-centric approach. This is important for any complex SoS represented as a mission engineering thread (MET) for both the primary path of execution, as well as the alternative paths that provide resiliency and redundancy in operations. A MET refers to an end-to-end sequence of steps that illustrate the technology, people, and resources needed to achieve a mission objective under specific conditions. It helps engineers understand how different systems interact within a SoS to accomplish the overall mission. In general, a MET provides insights into the functions, players, and interactions involved in a mission, the flow of data and decision-making across different systems, the impact of environmental factors on mission performance, and the resilience and availability of mission-critical systems.

Since the combination of these MET represents the chain of operational activities producing effects for certain operational capability, they will be referred to as 'effects webs,' as opposed to for example kill chain, to capture different kinds of graduated effects, such as distract or degrade that do not necessarily always end with destruction. To deal with different taxonomies and ontologies across industrial sectors and military services, a common framework or reference architecture is employed so that a logical comparison can be made across ME products (ref. Figure 2).

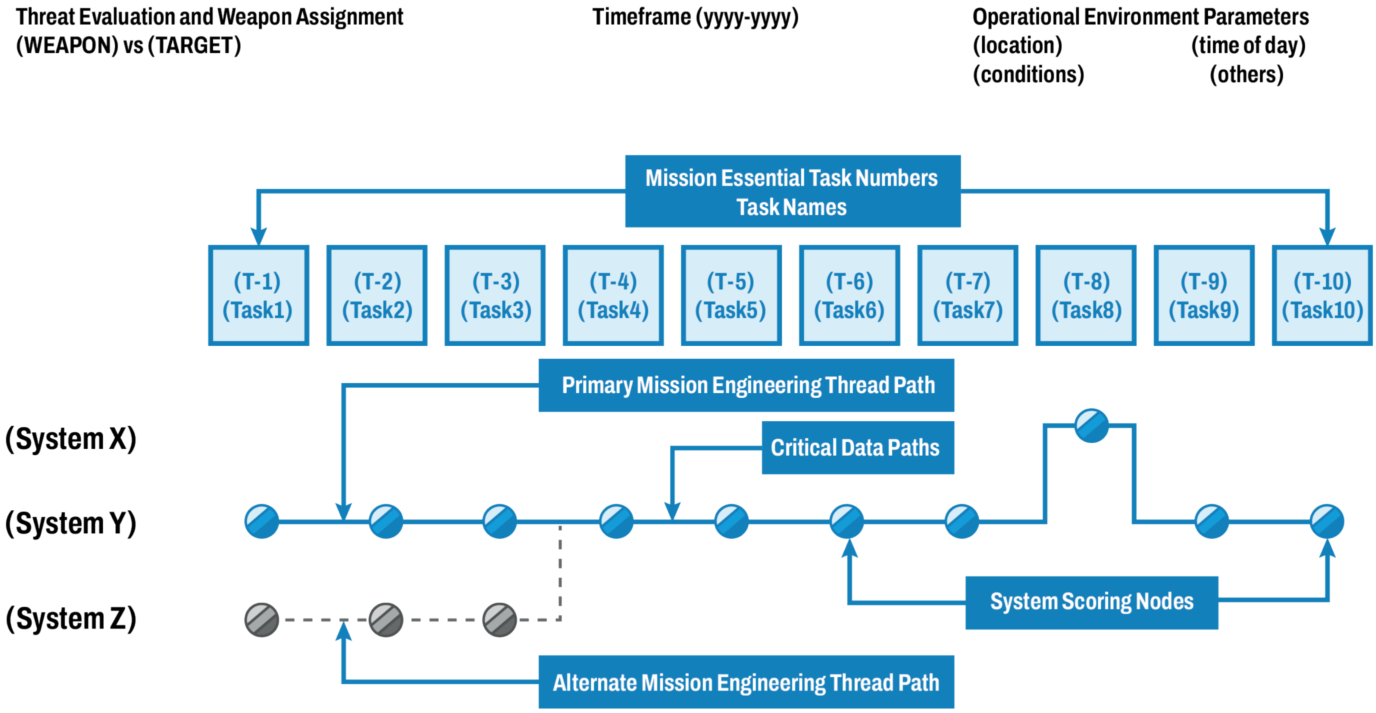


Figure 2. Generic effects web framework providing common terminology [2].

This operational approach is also valid for business applications, where they may define their own desired effects and corresponding METs and effect webs.

2. WHY ME AS A NEW APPROACH AND DISCIPLINE

Advances in adversarial (or competitors') capabilities pose a significant challenge to security (or business success), as new threats that can range from next-generation weapon systems and cyber warfare tactics to artificial intelligence-driven decision-making (or other novel technologies) are fielded. Any of these types of technologies, as well as many others, can rapidly shift the balance of power (or market). These advancements enable adversaries (or competitors) to operate with increased competitiveness (like reach, persistence, speed, precision, and lethality in defense contexts), potentially eroding strategic advantages and creating vulnerabilities in critical defense systems (or product/market portfolio in commercial sectors).

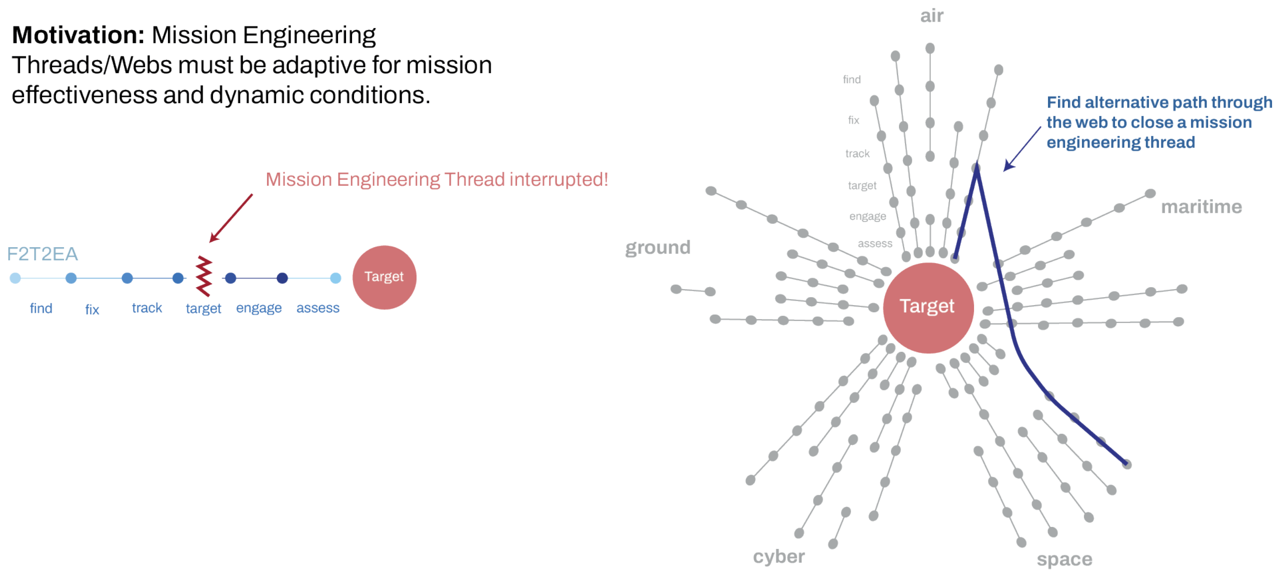
A broken MET occurs when there are disruptions or failures in the sequence of activities and systems required to achieve a mission objective. This situation leads to a failure in an OA and to a compromised mission effectiveness. This eventually translates into a gap or shortfall in capability. This breakdown can be caused by various factors, such as technical malfunctions, cyber-attacks, inadequate integration between systems and processes, logistics failures, communication breakdowns, or supply chain disruptions. For example, insufficient fuel or ammunition delivery, degraded satellite communications, or delays in troop movement can all sever critical links in mission execution.

To mitigate the risks associated with broken METs, redundancy and resiliency are generally embedded into mission planning and execution. Redundancy ensures that alternative pathways and backup systems are available to perform critical functions to maintain the continuity of operations when primary systems fail. Resiliency focuses on the capacity to adapt and recover quickly from disruptions. It enables missions to withstand and bounce back from unexpected challenges. Together, redundancy and resilience create a robust operational architecture that safeguards mission success even under adverse conditions, reinforcing the integrity and reliability of METs.

The ME approach prompts us to consider the potential mission thread paths using the Effects Web Framework based on the execution of defined mission essential tasks (e.g., how the force plans to fight in a military context). Defining this framework provides a common mechanism to decompose the mission into essential functions –also called functional activities (FA) in the NATO framework– so that joint/coalition systems can be mapped in a logical manner. Otherwise, when all joint/coalition forces are allowed to bring their own framework and terminology to the assessment table, performing a real function-to-system mapping across the entire force is highly challenging or even infeasible. In the past, this has amounted to integration and interoperability problems rendering the MET broke and the lack of a SoS capability for specific missions coming mainly from the isolated application of FA to standalone systems that, in the end, did not completely fulfill an operational capability.

End-to-end mission assessments are performed to grant a SoS resistant and resilient. They are based on operational evaluations that start with capability and spread OA and FA across the entire SoS, while it boosts the integration and interoperability of multiple systems to ensure accuracy in determining capability gaps. This mission-wide comprehensive approach provides data-informed decision-making that is not only valid to provide materiel solutions for operational gaps or shortfalls but also posits solutions across the whole structural capability definition spectrum (DOTMLPF-P for USA, DOTMLPF-I for EU, MIRADO for Spain), considering both materiel and non-materiel areas. Moreover, even when applied only to materiel solutions, the proposed approach brings better advantage when holistic technological solutions comprising a SoS are required.

Understanding the structural capability definition spectrum is critical when developing a materiel solution because it ensures that the solution is not only technologically viable but also fully integrated into the broader operational framework. A new capability must align with doctrine, be supported by organizational structures, incorporate effective training programs, and account for personnel requirements, infrastructure needs, and policy constraints. Ignoring any of these factors can lead to a materiel solution that is ineffective, unsustainable, or incompatible with existing systems, ultimately reducing mission effectiveness. Understanding SoS at this level of detail reduces the risk of fielding broken METs as you consider solutions across operational domains and industrial sectors. Figure 3 provides an illustration of a MET across the defined warfighting or operational domains.



Motivation: Mission Engineering Threads/Webs must be adaptive for mission effectiveness and dynamic conditions.

Mission Engineering Thread interrupted!
F2T2EA
find
fix
track
target
engage
assess
Target
air
find
fix
track
target
engage
assess
Find alternative path through the web to close a mission engineering thread
maritime
ground
Target
cyber
space

Figure 3. Multi-domain Mission Engineering Thread (MET).

In addition, this approach allows us to pay attention to resilient alternatives when executing METs while reducing unnecessary redundancy within and across the military services. Figure 4 represents the case of working across warfighting or operational domains while looking at alternative paths of mission execution that provide resiliency and redundancy. In order to take advantage of the methodology, it is important to ensure that formal analysis is done to assess the integration and interoperability of SoS required to execute mission threads. Formal analysis and assessment of SoS integration and interoperability to execute mission threads drives the development and implementation of today's US DoD ME approach. Analysis using the Effects Web Framework identifies operational gaps, science and technology solution insertion points, concepts of employment, and research and development requirements for the future solution space and "to-be" architecture across mission areas.

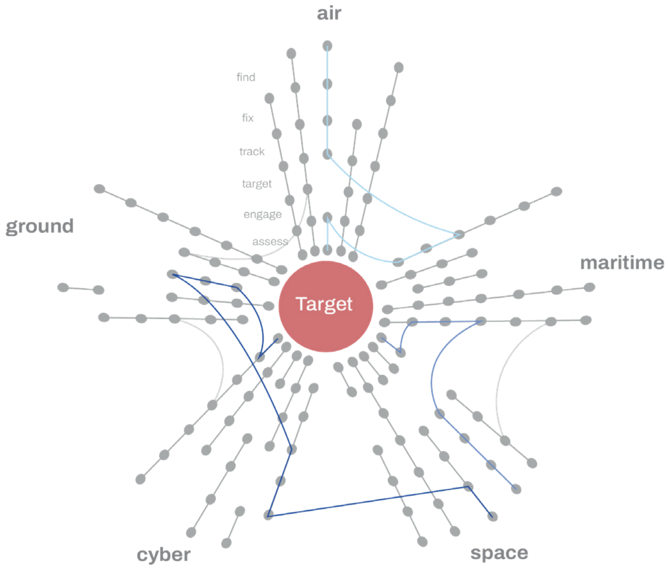


Figure 4. Multi-domain Mission Engineering Web.

Effective mission risk management is critical throughout the defense acquisition process to ensure that systems meet operational requirements and maintain mission effectiveness. This is relevant to any industry deploying critical capabilities or business delivering systems that are critical to their operations. When assessing mission risk, evaluators must consider the mission itself, including the unit equipped within the system, the operationally relevant environment, vulnerabilities of the system against the full spectrum of expected threats, and their combined effects in the context of its intended operational missions. In DoD programs, Mission-Based Risk Assessments (MBRAs) are being integrated into acquisition planning following an ME-based approach to evaluate mission-critical functions against potential threat effects in operationally representative scenarios (see Figure 5). These assessments identify vulnerable system elements and interfaces, informing the scope of operational test and evaluation, including live fire test and evaluation, to ensure systems can withstand real-world threats. By conducting MBRAs, program managers can monitor and quantify risks to test objectives, acquisition programs, end users, and overall UD DoD operations. This risk-based approach ensures that testing strategies are appropriately scaled and focused, facilitating informed decision-making and enhancing the resilience and effectiveness of defense systems in complex operational environments.

When METs break, the resulting rework and loss of capabilities can significantly degrade mission readiness and national security. Critical assets may need to be reallocated, delayed, or entirely redesigned, leading to wasted resources, increased costs, and operational setbacks. These failures can also create gaps in capability, reducing force effectiveness and leaving vulnerabilities that adversaries may exploit. The cascading effects of these disruptions can hinder strategic objectives, delay response times, and ultimately weaken the nation's ability to project power, defend interests, and maintain a technological edge in an evolving threat landscape.

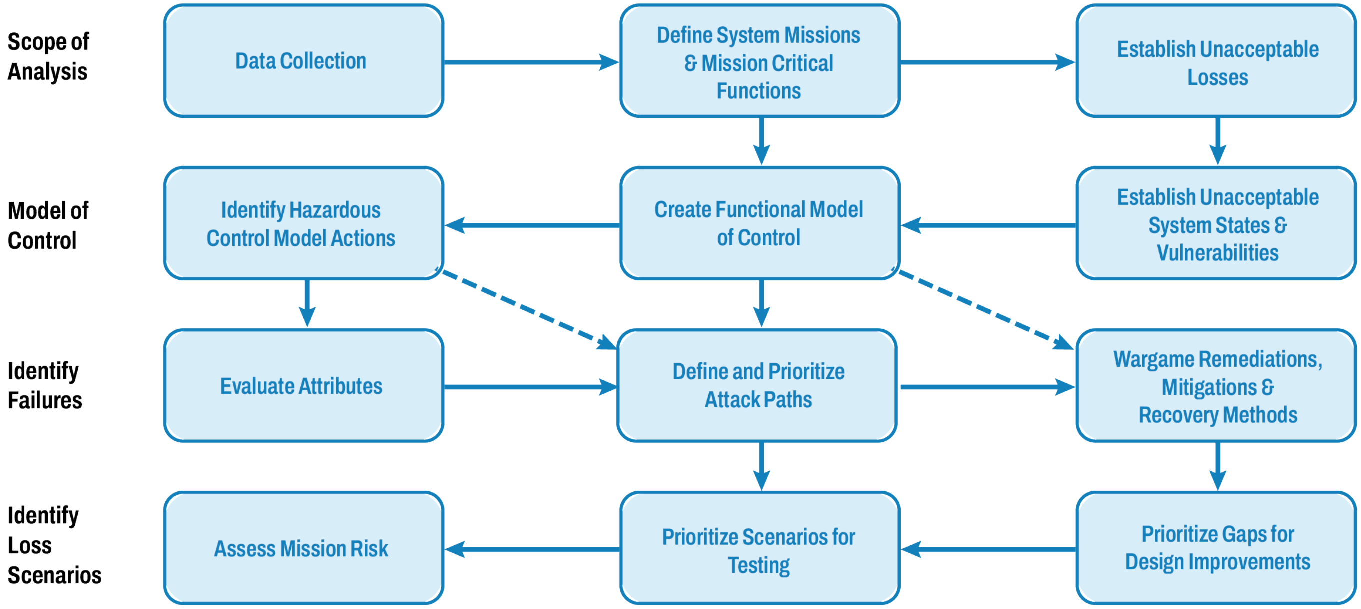

Scope of Analysis
Data Collection
Define System Missions & Mission Critical Functions
Establish Unacceptable Losses
Model of Control
Identify Hazardous Control Model Actions
Create Functional Model of Control
Establish Unacceptable System States & Vulnerabilities
Identify Failures
Evaluate Attributes
Define and Prioritize Attack Paths
Wargame Remediations, Mitigations & Recovery Methods
Identify Loss Scenarios
Assess Mission Risk
Prioritize Scenarios for Testing
Prioritize Gaps for Design Improvements

Figure 5. Mission-based risk management.

3. VALUE PROPOSITION

The assessment of US DoD technologies, systems and/or capabilities requires a SoS approach to analyze the impact of making these complex investments across the diverse domains of surface, undersea, air, ground, space, cyber, and networks as well as coalition force integration. In today's acquisition environment, programs are far too often matured independently and SoS integration occurs when delivered to the field rather than during early development, which increases cost and time to introduce a new capability to the end user. Mission engineering emphasizes capability-based assessments to produce integrated warfighting capabilities that can be translated into specific programmatic guidance to translate operational needs into SoS and system requirements to drive today's readiness and the future capabilities of the individual military services. The approach also provides critical insights into operational gaps to help inform stakeholders as well as allows us to make sure the right investments are being made across relevant organizations.

The complex and now highly integrated machines of operations continue to evolve enabling higher precision, more effective power projection, and safer defensive postures for the military services. The interconnectedness of our own social fabric is finding its way into our ships, aircraft, submarines, networks, space assets, tanks, and the very weapons they deliver. For example, determining the right investments for the development of space assets and the protection of positioning, navigation, and timing applications is extremely complex, requiring this systematic approach to identify all the major interfaces and intersection points amongst many systems/platforms in a multi-domain, multi-functional environment. Understanding SoS at this level of detail provides the opportunity to reduce the risk of fielding broken mission threads/capabilities as we look across warfighting or operational domains and individual military services. In addition, this approach allows us to pay attention to resilient alternatives for the execution of critical mission threads while reducing unnecessary redundancy within and across the joint/coalition force based on the way our forces plan to fight with integration and interoperability factors in mind.

4. ESSENTIAL ASPECTS OF MISSION ENGINEERING

4.1. Continuity through established mission success criteria

The SoS required to execute operational missions are too often 'stovepipe-optimized' without sufficient deference to total SoS operational utility. ME analyses often leverage simulations and other tools to develop Measures of Success (MoS), Measures of Effectiveness (MoE), and Measures of Performance (MoP) for the constituent systems while assessing the mission success criteria for the mission essential tasks to achieve overall mission objectives through experimentation. Something that is also applicable to business.

Linking mission effectiveness from strategic to operational to tactical levels requires a structured approach that aligns overarching goals with actionable and measurable criteria at each level. At the strategic level, organizations define MoS that articulate the desired end state and long-term impact of the mission. These high-level criteria serve as guiding principles that ensure all subsequent efforts contribute to overarching objectives, such as national security or strategic autonomy, economic stability, or regional influence. Strategic leaders determine MoS by assessing factors like deterrence capability, global presence, or force readiness, ensuring alignment with policy objectives and broader strategic imperatives.

At the operational level, these strategic success measures are translated into MoE that assess how well mission essential tasks (often structured as a mission thread) contribute to achieving the strategic goals. MoEs provide a way to evaluate whether key operational actions are contributing to desired outcomes, focusing on effectiveness rather than efficiency. For example, in a military context, MoEs for an air superiority mission might include sustained control of designated airspace, response time to emerging threats, or enemy attrition rates. These measures ensure that commanders can adjust tactics, resource allocations, and mission execution in response to real-time conditions while maintaining alignment with strategic intent.

Lastly, at the tactical level, MoP are established to evaluate how well individual systems, personnel, and processes function in executing mission essential tasks. MoPs are quantitative and focus on system-specific capabilities, such as sortie generation rates for aircraft, readiness state, weapon accuracy, or the uptime of critical communication systems. These measures provide direct feedback on whether the assets and processes at the tactical level are performing at the required standards to support operational effectiveness.

In brief, MoEs criteria measure if we are doing the right things and the MoPs criteria measure if we are doing the things right. By continuously monitoring MoPs, tactical units can make necessary adjustments, ensuring their performance directly contributes to mission success at higher levels. MoPs are traditionally written and referred to as Key Performance Parameters and Key System Attributes as these are the criteria that are built into requirements documents for the design or alteration of systems and where the design trade space occurs. The linkage across these three levels ensures that strategic goals drive operational planning, and operational effectiveness depends on tactical execution. By clearly defining MoS, MoE, and MoP, organizations create a coherent framework that aligns decision-making, resource allocation, and performance assessment, ultimately enhancing overall mission success.

Figure 6 shows a representative mapping of these scoring criteria levels which exists to create continuity across all levels of operations with the goal of achieving unity of effort towards the defined mission success.

Effect Based Operations (EBO) are an example of how the indicators and metrics transformed military operations after the Gulf War I. Previously, the attrition of adversarial assets was the standard. The objective was then to eliminate enemy forces in the search for a withdrawal or surrender, but the EBO approach relied upon the basis of the analysis of the enemies' Centers of Gravity (CoG) in order to design plans to deny those centers under the minimum cost premise. EBO are therefore based on effects, and those effects are provided by operational activities that in the end, are the more atomic functional elements of a capability as defined by the NATO. Effects are, hence, sustained by the capabilities, linking the entire landscape of the levels of operational execution, from strategical to tactical. Since CoG are currently dispersed across the entire mission arena, the SoS is the enabler to synchronize the achievement of combined effects to make them valuable for the operation. ME, follows this approach to extend this link to the technical level.

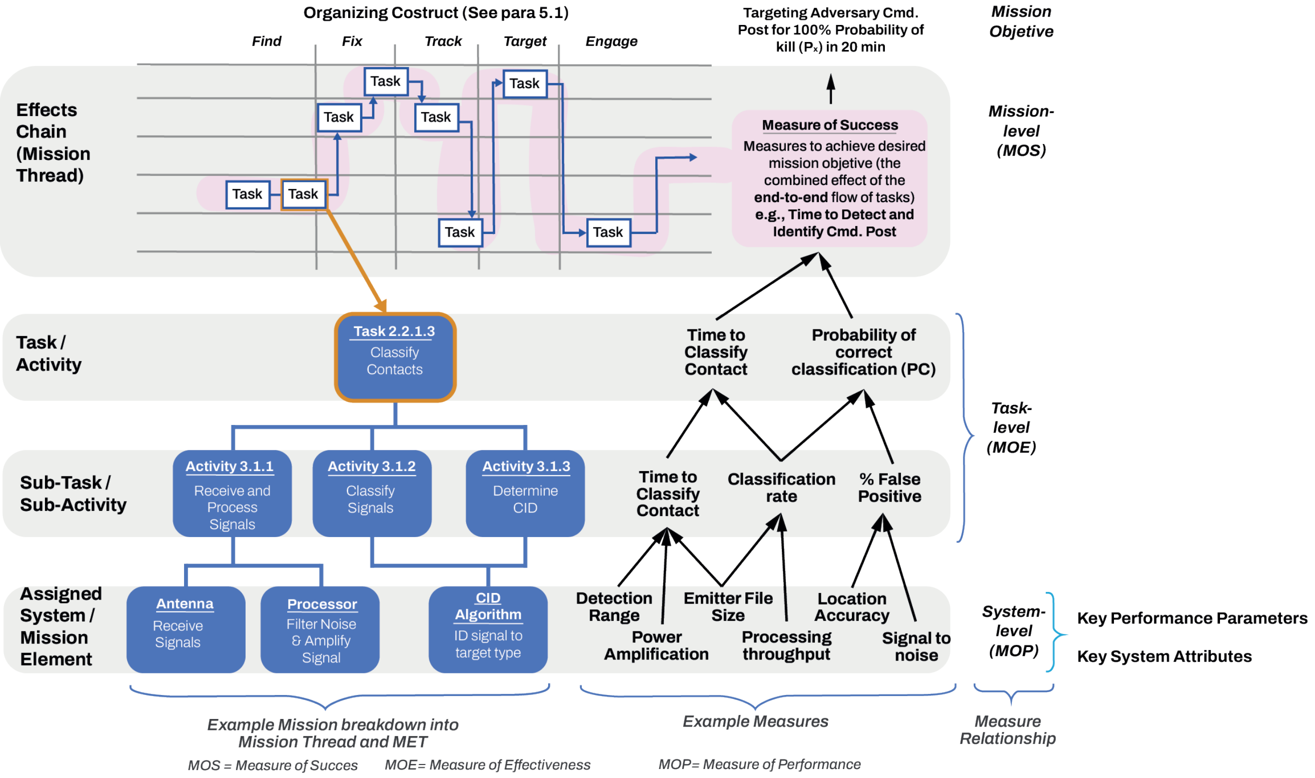


Figure 6. Developing Mission success criteria at all levels of operation.

In today's execution of acquisition planning processes, systems are designed and developed based on the specific need to meet a requirement, not necessarily to support SoS effects. This is typical of the acquisition process through the Joint Capability Integrated Development System, which is designed to develop a system within a specific context rather than across a domain or integrated mission-based context. The individual systems are not developed with the idea of integrating across a SoS architecture to execute operational METs. In other words, they are not capability-focused programs pursuing the achievement of operational remarkable effects at SoS level. Program managers understandably optimize product design based on their specific customer needs without assessing how the system interacts with other deployed systems to collaboratively support the desired effects during the execution of critical mission threads. This 'intra-optimal' stovepipe system design is sub-optimal when later evaluated from an 'inter-optimal' SoS perspective. Unfortunately, considerable amounts of redesign would be required to modify the system with respect to inter-optimal SoS mission success. Engineering and deploying products with SoS mission success in mind during system design is the way to avoid integration and interoperability issues across the force (see Figure 7).

An uncontrollable challenge is that the number of analytic scenarios required to support robust SoS design increases exponentially with the addition of each operational mission, performance parameter, design criteria, adversarial threat, and environmental factor. This combinatorial explosion challenges the use of many traditional processes and tools for realistic ME design of complex SoS. There is a need to develop the principles, processes, and tools to assess and support the design of complex SoS using the ME approach to make the design of complex SoS more tractable. With that said, the principles and foundations of systems engineering are necessary but not sufficient to handle these complexities of today.

While JCIDS (Joint Capability Integration and Development System) provides a structured acquisition approach within the US DoD, no direct equivalent currently exists in Europe. Frameworks such as the NATO AAP-20 or national approaches (e.g., Spain's MIRADO-I) are still under development and differ substantially in maturity and governance.

A mission-based engineering approach helps ensure the system under development will easily integrate with other systems while ensuring the capabilities necessary for mission success. Further, the acquisition system is not equipped with the ability to perform governance across a SoS; this means that when ME is performed it is unlikely to remain stable for a duration longer than one of the systems' upgrade or maintenance cycles. This opportunistic approach to ME does not necessarily provide long term stability to support missions but puts the stakeholder on the right vector for functional effectiveness from the start. As mentioned previously, the acquisition approach to concept, design, and experimentation would need to be developed to ensure the governance across any SoS could be guaranteed through synchronization which is the current direction with capability portfolio management.

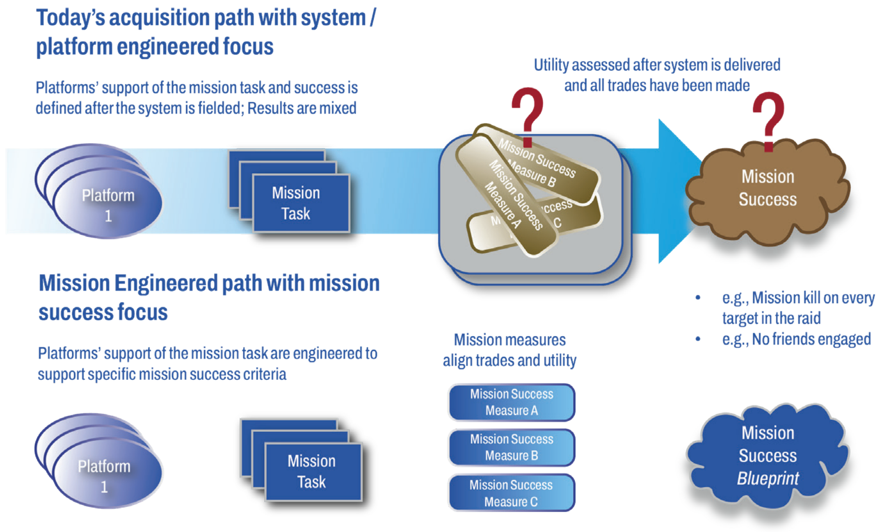
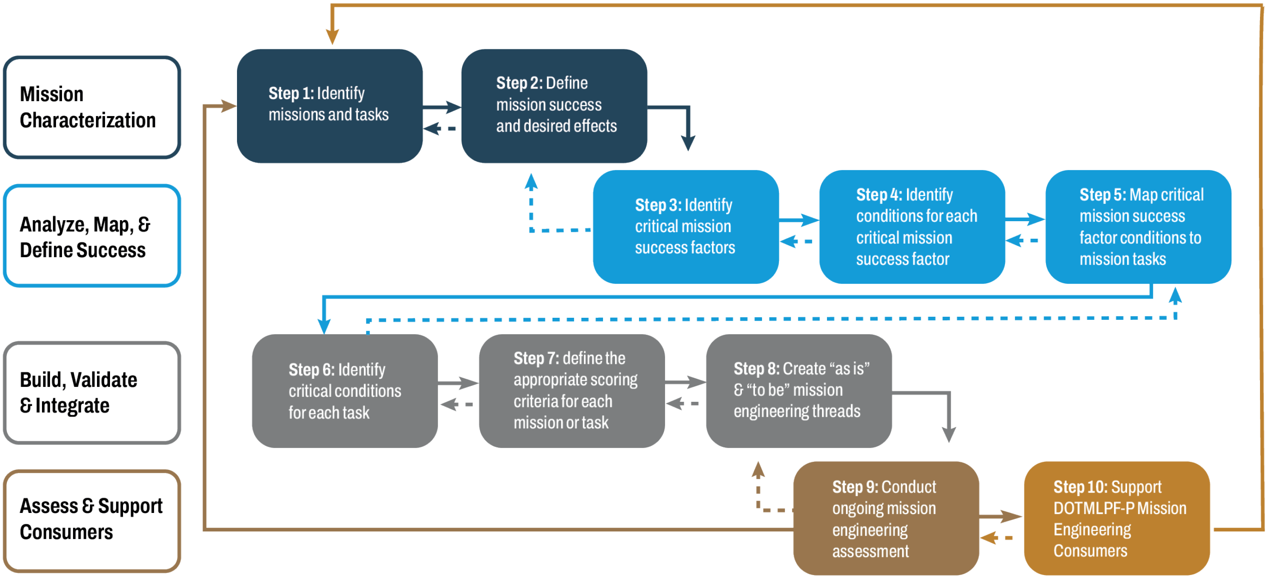

Today's acquisition path with system / platform engineered focus
Platforms' support of the mission task and success is defined after the system is fielded; Results are mixed
Platform 1
Mission Task
Utility assessed after system is delivered and all trades have been made
?
Mission Success Measure B
Mission Success Measure A
Mission Success Measure C
?
Mission Success
Mission Engineered path with mission success focus
Platforms' support of the mission task are engineered to support specific mission success criteria
Platform 1
Mission Task
Mission measures align trades and utility
Mission Success Measure A
Mission Success Measure B
Mission Success Measure C
e.g., Mission kill on every target in the raid
e.g., No friends engaged
Mission Success Blueprint

Figure 7. Aligning operational Mission SoS to defined Mission success criteria.

4.2. Disciplined approach to field capabilities

The structured outlined approach to ME found in MEG v2.0 can be enhanced by following the 10-step process introduced in this section. This process is mapped to the MEG v2.0 approach categories as shown in Figure 8.

This process begins with identifying missions and tasks and progresses through defining mission success criteria, critical success factors, and associated conditions that impact performance. By mapping these conditions to mission tasks and developing clear scoring criteria, decision-makers can evaluate current capabilities against future needs, creating "as-is" and "to-be" mission engineering threads to guide improvements. Ongoing ME assessments ensure adaptability in dynamic environments, while alignment with DOTMLPF-P (DOTMLPF-I in the EU, MIRADO-I in Spain) ensures that ME efforts support broader force development. Through this structured approach, ME provides a data-driven foundation for capability development, risk mitigation, and operational effectiveness, ensuring that mission planning and execution remain resilient against emerging threats.



Mission Characterization
Analyze, Map, & Define Success
Build, Validate & Integrate
Assess & Support Consumers
Step 1: Identify missions and tasks
Step 2: Define mission success and desired effects
Step 3: Identify critical mission success factors
Step 4: Identify conditions for each critical mission success factor
Step 5: Map critical mission success factor conditions to mission tasks
Step 6: Identify critical conditions for each task
Step 7: define the appropriate scoring criteria for each mission or task
Step 8: Create "as is" & "to be" mission engineering threads
Step 9: Conduct ongoing mission engineering assessment
Step 10: Support DOTMLPF-P Mission Engineering Consumers

Figure 8. ME 10-step process mapped to MEG v2.0.

The order of the 10 ME steps is outlined below starting with the prioritization of operational mission areas and ending with the continuous management of end-to-end METs to maintain the execution health of capabilities.

1) Identify missions and tasks.
2) Define mission success and desired effect.
3) Identify critical mission success factors.
4) Identify conditions for each critical mission success factor.
5) Map critical mission success conditions to mission tasks.
6) Identify critical conditions for each task.
7) Define the appropriate scoring criteria for each mission or task.
8) Create "as-is" and "to-be" mission engineering threads.
9) Conduct ongoing mission engineering assessments.
10) Support DOTMLPF-P (DOTMLPF-I in the EU, MIRADO-I in Spain) Mission Engineering Consumers.

The implementation of this process comes with challenges associated with governance structure, data availability and collection, stakeholder coordination across the US DoD, multiple system life cycles due to maturation (legacy to new), and workforce/tool development to name a few. However, the Effects Web Framework (Figure 2) provides a mechanism to translate what the US DoD plans to procure to the resulting capability. An example outcome of this process showing the potential SoS capability as an assessed MET is illustrated in Figure 9.

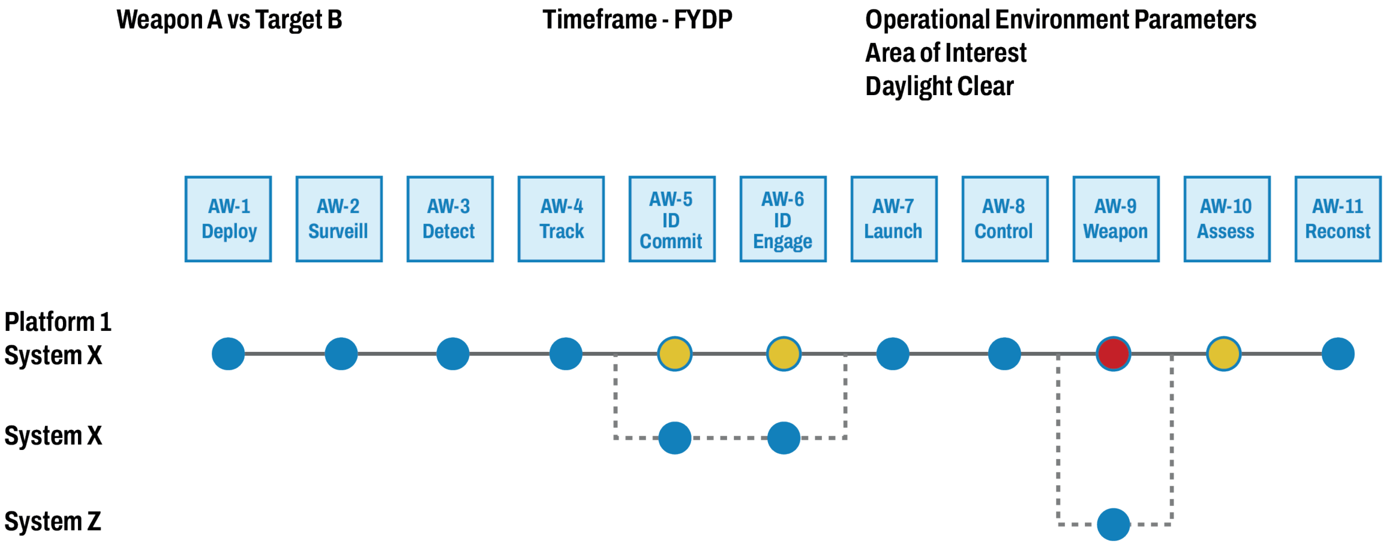


Figure 9. Scoring criteria applied for example.

4.3. Opportunity to integrate capabilities across nations

Our nations are becoming more complex every year as we move towards fielding the 'Internet of Things' and work towards fielding 'Digital Twins' for more connectivity and monitoring of performance health. The lack of common approaches to analyze these complex SoS with new tools and processes will continue to cause delays in our ability to execute rapid development and fielding of capabilities. In fact, the ME approach has tremendous promise in bringing together the international community of allied partners as we prepare to fight as a joint/coalition force (see Figure 10). This figure shows only allied partners who have been in involved in recent discussions with the US DoD; however, this does not represent a bounded state on bringing in other allied partners into the discussions to increase coordination and collaboration using ME principles and processes.

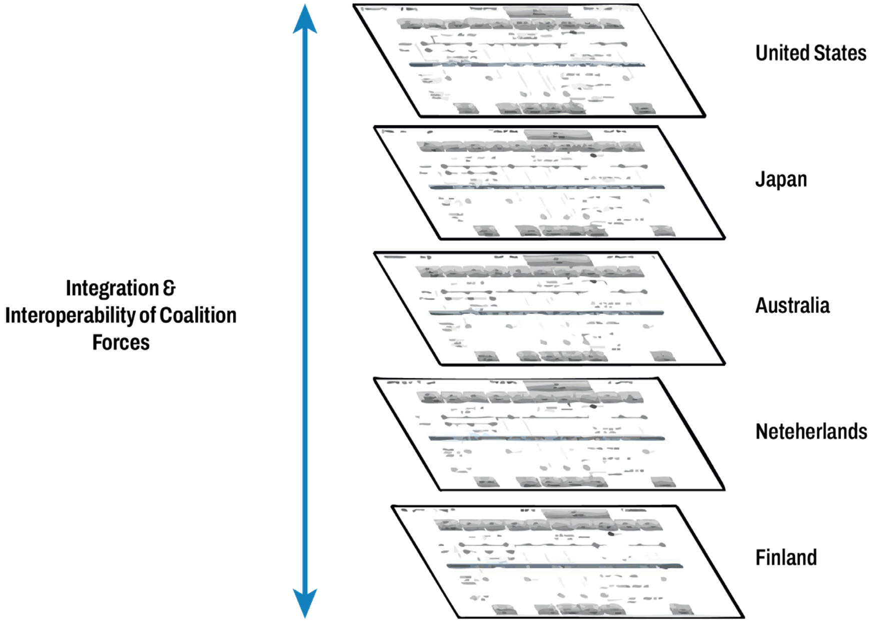


Figure 10. Vertical integration and interoperability across allied partners.

While ME is maturing rapidly within the US DoD, European institutions such as the European Defence Agency (EDA), Spain's Directorate General of Armament and Materiel (DGAM), and national frameworks like MIRADO are progressively integrating ME principles into their planning and capability development processes. Increased collaboration between US and European ME initiatives could help create shared methodologies, interoperability standards, and joint SoS assessments.

7. CONCLUSIONS

Mission Engineering provides a strategic approach to designing and managing complex SoS that extend beyond traditional defense applications. While critical for national security and strategic autonomy, ME also enhances the ability to address a range of mission challenges, including disaster response, humanitarian assistance, infrastructure resilience, and space exploration. By leveraging principles such as interoperability, open systems architecture, and risk-informed decision-making, an organization or a nation can optimize resources and improve mission outcomes across multiple domains.

The application of ME in civil-military cooperation, emergency management, and technological innovation ensures that an organization remains agile in responding to both anticipated and unforeseen challenges. Integrating human systems considerations, sustainability factors, and cross-sector collaboration further strengthens the country's capacity to develop solutions that balance operational effectiveness with long-term societal benefits. Ultimately, ME enables an organization to proactively shape its mission capabilities in a way that is adaptable, cost-effective, and technologically advanced. By fostering an approach that spans defense, security, and broader national priorities, an organization can enhance its strategic resilience while contributing to stability and progress on both national and global scales.

REFERENCES

1. OSD R&E. (2023, October). Mission Engineering Guide (MEG) v2.0. Retrieved from https://ac.cto.mil/wp-content/uploads/2023/11/MEG_2_Oct2023.pdf.

2. Moreland Jr., James D. (2014). "Mission Engineering Integration and Interoperability." NSWCDD Leading Edge Technical Digest, NSWCDD-MP-15-00096 01/15: pp. 4-15.

BIOGRAPHIES

DR. JAMES MORELAND JR.

Dr. James Moreland Jr. retired from the Senior Executive Service at the U.S. Department of Defense in 2020 and subsequently served as Vice President of Strategy and as Vice President of Mission Engineering and Integration at Raytheon Technologies, concluding five years of service with the company in April 2025. Prior to that, he served as the Executive Director - Mission Engineering and Integration, Deputy Assistant Secretary of Defense for Tactical Warfare Systems, and Executive Director - Naval Warfare in the Office of the Secretary of Defense, and as Chief Engineer of the Naval Surface Warfare Center, Dahlgren Division. He received a B.S. in Mechanical Engineering from the University of Maryland, M.S. in Systems Engineering from Virginia Tech, M.S. in National Resource Strategy from the Industrial College of the Armed Forces, and a Ph.D. in Systems Engineering from the George Washington University. He has served as a senior executive advisor to the US National Science Board and White House Office of Science and Technology and will be serving on the US Air Force Scientific Advisory Board starting in September 2025.

LT. COL. VÍCTOR M. SOBRINO

Lt. Col. Víctor M. Sobrino is an Air Force F-18 pilot currently assigned to the European Defense Agency as Project Officer for the European Command and Control Program. Before this assignment he was the Operational Head at the Next Generation Weapons System Spanish Program Office. He was Squadron commander in the 12th Fighting Wing of the Spanish Air force, and a staff officer in the Spanish Air Force Air Combat Command as well as in the Joint Force Air Component (JFAC). Apart from his military studies that includes the Spanish Armed Forces Joint Major Staff Course, he received a B.S. in Computer Science, a M.S in Advanced Artificial Intelligence, a Master in Business Administration, and a Master in Direction and Management of Defense Procurement Systems. He is currently pursuing a Ph.D. In artificial Intelligence from the Polytechnical University of Madrid (UPM) and the Spanish National Cancer Research Center (CNIO).

Last Update 15:24
Date Analysing...
GMT +1

CHAPTER 4

Mission architecture modeling

Matthew Gagliardi, *System Strategy, Inc. (SSI) (MGagliardi@Systemxi.com)*
Matthew C. Hause, *System Strategy, Inc. (SSI) (MHause@Systemxi.com)*
Dr. James N Martin, *The Aerospace Corporation (James.N.Martin@aero.org)*
Víctor Ramos del Pozo, *Isdefe (vramos@isdefe.es)*

Abstract

When planning and conducting a Mission Engineering (ME) study, it is important to have a comprehensive, accurate, and coherent model of the mission architecture. This chapter explores some of the key modeling features and constructs that enable the development of a mission architecture model to be used in support of an ME study effort. It discusses extensions being applied to the Unified Architecture Framework (UAF) to better support ME activities.

Keywords

Mission Engineering, Mission Architecture, Mission Threads, Mission Engineering Threads, Enterprise Architecture, Unified Architecture Framework

1. INTRODUCTION

A mission architecture typically involves multiple enterprises and complex relationships between the enterprise entities (including the systems of systems, SoS, and individual systems). The Unified Architecture Framework (UAF) works well in capturing many of the "non-physical" aspects of the relevant systems of systems, since it highlights both materiel and non-materiel (such as, doctrine, organization, training, leadership, personnel, facilities) solutions that are involved in a mission architecture. UAF specifies a set of architecture views for describing various aspects of an enterprise and major entities in the enterprise [1-3] and provides a modeling language (UAFML) that is especially designed for modeling an enterprise. As such, it is appropriate for modeling a large and complex mission architecture along with its variety of scenarios, vignettes, MTs, METs, etc. [4, 5]. Some of the structural model elements relevant to mission modeling are illustrated in Figure 1 and described in Table 1.

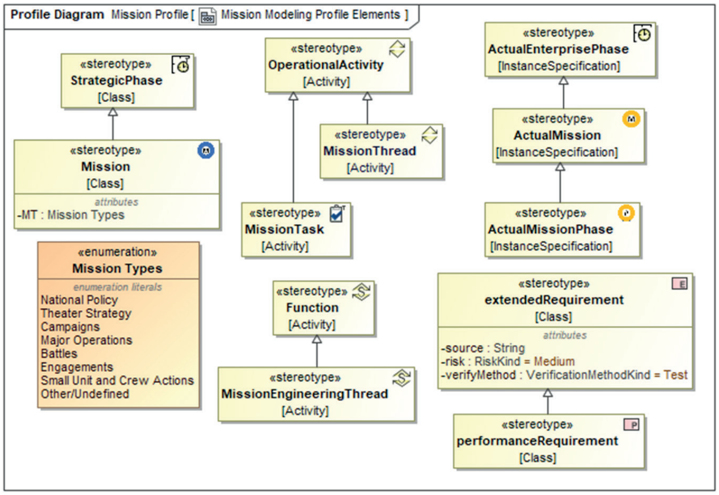

Profile Diagram Mission Profile [Mission Modeling Profile Elements]
«stereotype» StrategicPhase [Class]
«stereotype» Mission [Class]
attributes
-MT : Mission Types
«enumeration» Mission Types
enumeration literals
National Policy
Theater Strategy
Campaigns
Major Operations
Battles
Engagements
Small Unit and Crew Actions
Other/Undefined
«stereotype» OperationalActivity [Activity]
«stereotype» MissionThread [Activity]
«stereotype» MissionTask [Activity]
«stereotype» Function [Activity]
«stereotype» MissionEngineeringThread [Activity]
«stereotype» ActualEnterprisePhase [InstanceSpecification]
«stereotype» ActualMission [InstanceSpecification]
«stereotype» ActualMissionPhase [InstanceSpecification]
«stereotype» extendedRequirement [Class]
attributes
-source : String
-risk : RiskKind = Medium
-verifyMethod : VerificationMethodKind = Test
«stereotype» performanceRequirement [Class]

Figure 1. Mission modeling profile view.

Modeling Element	Description
Mission	A Mission element is a generalization of an Enterprise Phase element in the UAF Domain Metamodel.
Actual Mission	An Actual Mission is a generalization of an Actual Enterprise Phase element in the UAF Domain Metamodel.
Actual Mission Phase	An Actual Mission Phase is a specialization of an Actual Mission providing an instance specification of a Mission and a Mission Phase.
Mission Thread	A Mission Thread is a generalization of an Operational Activity.
Mission Task	A Mission Task is also a generalization of an Operational Activity, with Mission Threads being made up of other Mission Threads or Mission Tasks.
Mission Engineering Thread	Mission Engineering Thread is a generalization of a Function and describes the implementation of Actual Mission Phases. Traceability between the MET and MT uses the standard UAF implements relationship.

Table 1. Modeling elements for use in a Mission architecture model.

While UAF provides a large number of potential architecture views, as shown in Figure 2, mission architecture modeling only needs a small subset of these views. The Mission Problem definition and the Mission Characterization aspects of the mission, along with the Mission Thread (MT) elements and views to be used in ME, map mainly to the Strategic and Operational viewpoints in UAF as illustrated in Figure 2 [6]. The Mission Engineering Threads (METs) are an implementation of the MTs, so these are primarily depicted in the Resources viewpoint. However, note that there are several other UAF viewpoints and their associated modeling views that could be readily used in an ME study and in related activities such as capability planning, enterprise portfolio management, annual budget formulation, program assessment and evaluation, system requirements development, etc.

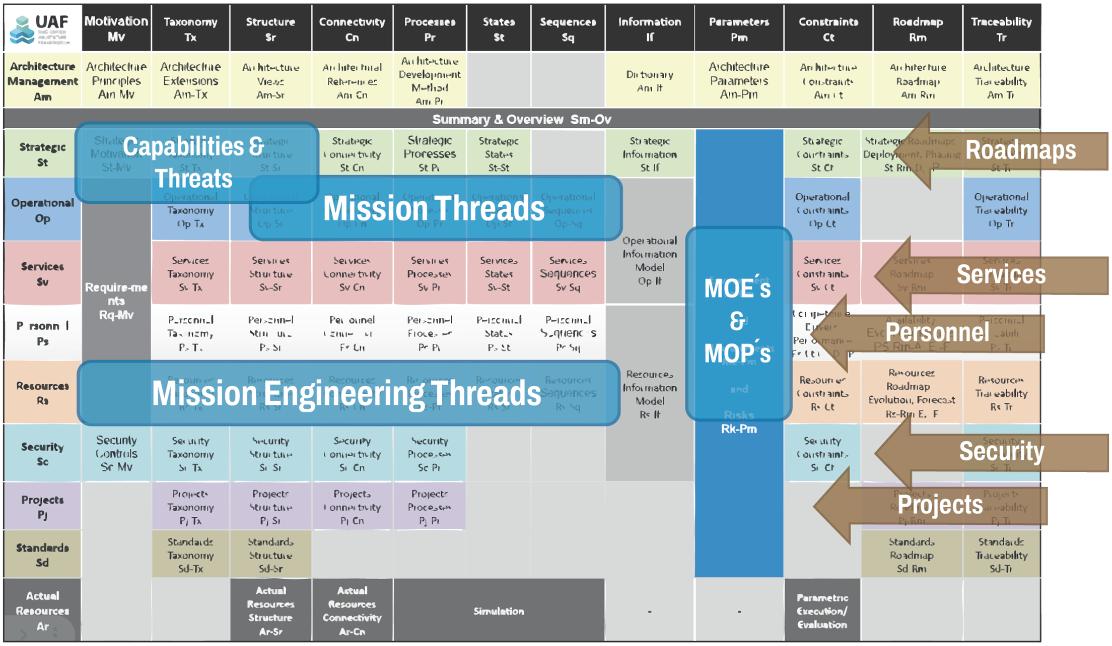


Figure 2. Mission engineering views in UAF.

To illustrate how the modeling language can be used to define a mission architecture, the next sections present the application of the ME concepts from the previous chapter on a mission exemplar using mission modeling.

2. MISSION EXEMPLAR

The mission exemplar used in this chapter is the Battle of Hoth from the second Star Wars movie, “The Empire Strikes Back.” This mission is used because it is well known, contains a rich source of systems, strategies, missions, and behaviors, and illustrates joint operations[1]. The example, adapted from [4, 5], concentrates on the Strategic and Operational views, defining the concepts of missions, mission phases, MTs and operational architectures, as well as definition of the resource and organizational structures and functionality.

The mission is defined as occurring in scenarios and vignettes over time, in a sequence of mission phases. The operational architecture, which is composed of several MTs and associated METs, is defined to satisfy motivational factors, such as the Drivers and Enterprise Goals shown in Figure 3. The operational architecture is used as the basis for operational effectiveness analysis, sometimes accomplished using modeling and simulation tools and techniques.

2.1. Mission purpose, stakeholders, concerns, goals, and drivers

Measures of Success (MOS), desired effects, and capabilities are captured in the mission architecture model, with the operational architecture elements tracing back to these elements to ensure proper coverage. Measures of Effectiveness (MOE) are assigned to the Mission Tasks that make up each of the Mission Threads.

Definitions for related key concepts are shown in Table 2 and the corresponding mission elements are shown in Figure 3. The Legion Commander is concerned about his loss of position or possibly his life, which typically happens when failure occurs in the service of the Empire. He wishes to prevent a rebel resurgence and to ensure a decisive victory. Darth Sidious and Darth Vader wish to control the Galaxy and establish dark side dominance. Darth Vader also wishes to protect Luke Skywalker. These Concerns relate directly to the mission goals, which then link to the Drivers which have forced the Empire to act. These will be discussed further on in the chapter.

[1] *For further background information see [7] or better yet, grab some popcorn and watch the movie.*

Concept	Description
Concern	A matter of relevance or importance to a stakeholder regarding an entity of interest.
Driver	Thing that forces to work or act; that which urges you forward.
Challenge	A demanding or stimulating situation; a call to engage in a contest or fight.
Enterprise State	Condition with respect to circumstances or attributes.
Capability	Ability to achieve a desired effect under defined conditions and environments.
Opportunity	A possibility due to a favorable combination of circumstances.
Risk	A source of danger; a possibility of incurring loss or misfortune.
Effect	A phenomenon that follows and is caused by some previous phenomenon.
Outcome	Something that happens or is produced as the final consequence or product.
Goal	A statement about a state or condition of the enterprise to be brought about or sustained through appropriate Means.
Objective	A statement of an attainable, time-targeted, and measurable target that the enterprise seeks to meet in order to achieve its Goals.

Table 2. Strategic motivation elements.

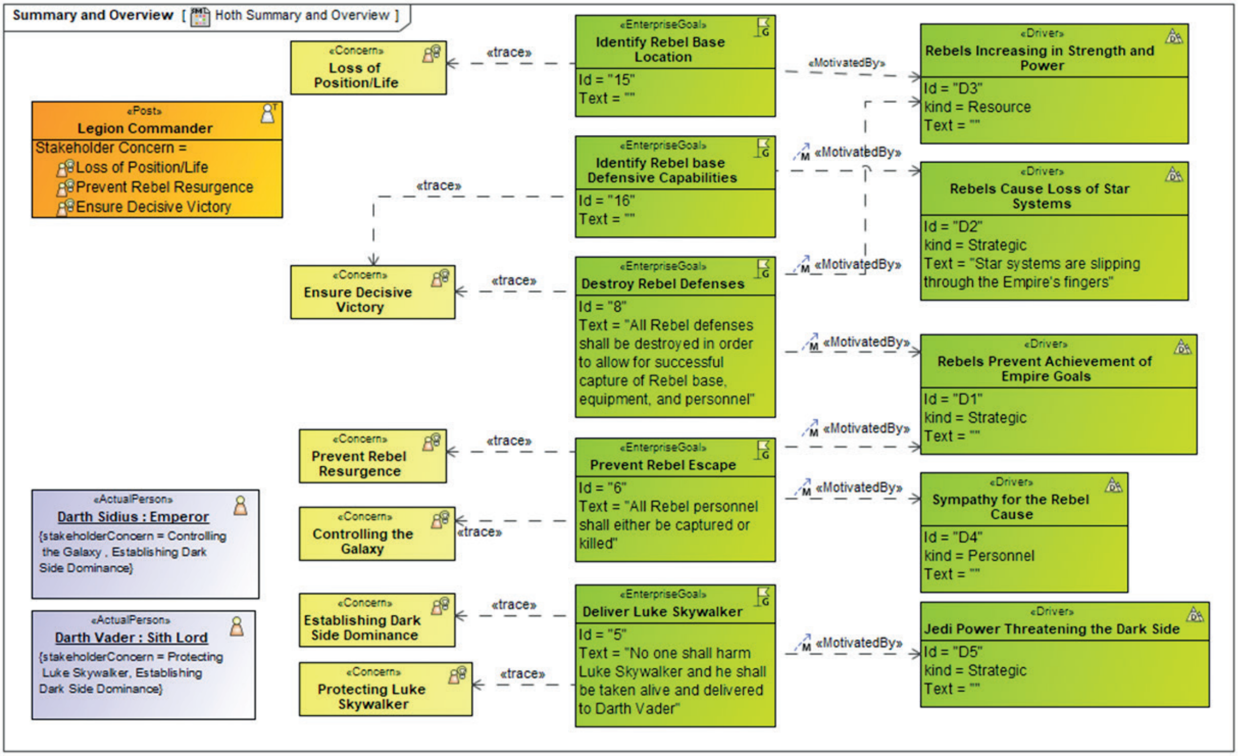

Summary and Overview [Hoth Summary and Overview]
«Post» Legion Commander
Stakeholder Concern =
Loss of Position/Life
Prevent Rebel Resurgence
Ensure Decisive Victory
«Concern» Loss of Position/Life
«Concern» Ensure Decisive Victory
«Concern» Prevent Rebel Resurgence
«Concern» Controlling the Galaxy
«Concern» Establishing Dark Side Dominance
«Concern» Protecting Luke Skywalker
«ActualPerson» Darth Sidius : Emperor
{stakeholderConcern = Controlling the Galaxy , Establishing Dark Side Dominance}
«ActualPerson» Darth Vader : Sith Lord
{stakeholderConcern = Protecting Luke Skywalker, Establishing Dark Side Dominance}
«trace»
«EnterpriseGoal» Identify Rebel Base Location
Id = "15"
Text = ""
«EnterpriseGoal» Identify Rebel base Defensive Capabilities
Id = "16"
Text = ""
«EnterpriseGoal» Destroy Rebel Defenses
Id = "8"
Text = "All Rebel defenses shall be destroyed in order to allow for successful capture of Rebel base, equipment, and personnel"
«EnterpriseGoal» Prevent Rebel Escape
Id = "6"
Text = "All Rebel personnel shall either be captured or killed"
«EnterpriseGoal» Deliver Luke Skywalker
Id = "5"
Text = "No one shall harm Luke Skywalker and he shall be taken alive and delivered to Darth Vader"
«MotivatedBy»
«Driver» Rebels Increasing in Strength and Power
Id = "D3"
kind = Resource
Text = ""
«Driver» Rebels Cause Loss of Star Systems
Id = "D2"
kind = Strategic
Text = "Star systems are slipping through the Empire's fingers"
«Driver» Rebels Prevent Achievement of Empire Goals
Id = "D1"
kind = Strategic
Text = ""
«Driver» Sympathy for the Rebel Cause
Id = "D4"
kind = Personnel
Text = ""
«Driver» Jedi Power Threatening the Dark Side
Id = "D5"
kind = Strategic
Text = ""

Figure 3. Hoth summary and overview: Stakeholder concerns and goals.

2.2. Mission definition

The Empire Mission structure shown in Figure 4 illustrates the complexity required to model missions. Empire doctrine proscribes that every military mission has two phases to it: Planning and Execution. A Planetary Invasion Mission is comprised of separate Scout, Landing, and Attack Missions, each with their own Planning and Execution Phases. These are all types of Invasion Missions. Each of these has a defined Mission Type. The Execution and Planning Phases both inherit Mission Tempo and Phase attributes. Mission type attributes have been defined for several of the mission types. Scenario and Vignette types have been linked to the missions that when instantiated define the parameters and context of the mission. These are detailed in the next section. Specific MTs can and should be linked to the various missions to define the functional aspects.

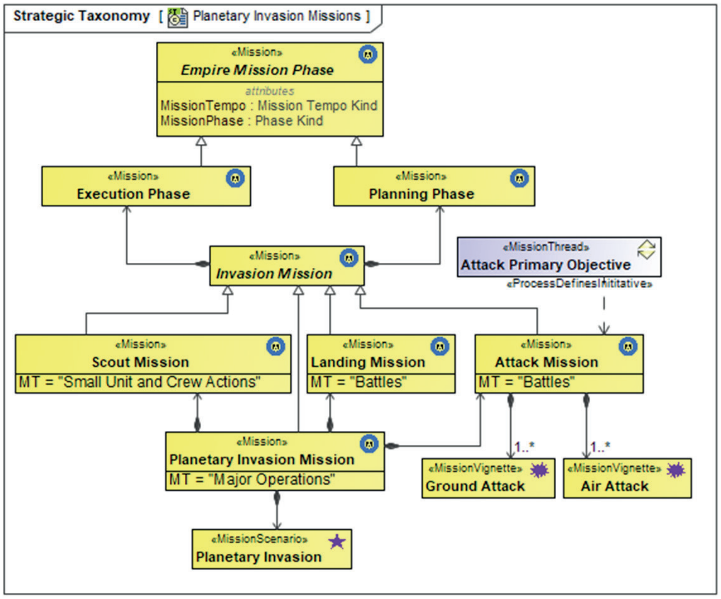


Figure 4. Mission definitions.

2.3. Campaign, scenario and vignette

Since mission names can include the mission type (e.g., Hoth Campaign, Hoth Scouting Operation, or Hoth Ground Battle), and the mission structural hierarchy can show what mission is comprised of other missions, mission types in Figure 5 are defined in an enumeration and modeled as an attribute for the Mission Types in Figure 3 instead of defining dedicated stereotypes. Values chosen for the Hoth example are shown in Figure 4.

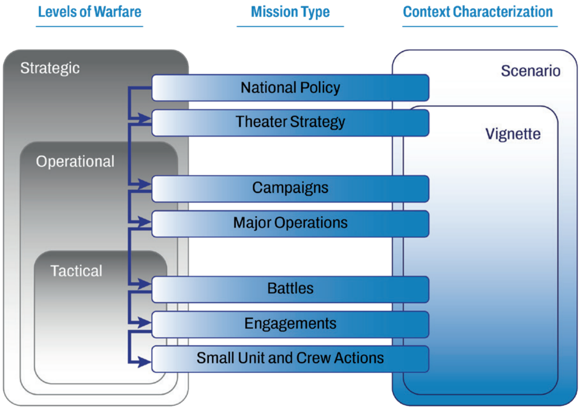


Figure 5. Levels of warfare and mission types [8].

For the concepts of Scenario and Vignette, new stereotypes are used to describe the necessary context information for the mission(s) being described in the model, as shown in Figure 6. Scenarios describe the geographical location and time frame of the overall conflict. They include information such as threat and friendly politico-military contexts and backgrounds, assumptions, constraints, limitations, strategic objectives, and other planning considerations [8]. Vignettes describe narrow and specific ordered sets of events, behaviors, and interactions for a specific set of systems to include blue capabilities and red threats within the operational environment. Vignettes can represent small, ideally self-contained parts of a scenario [8].

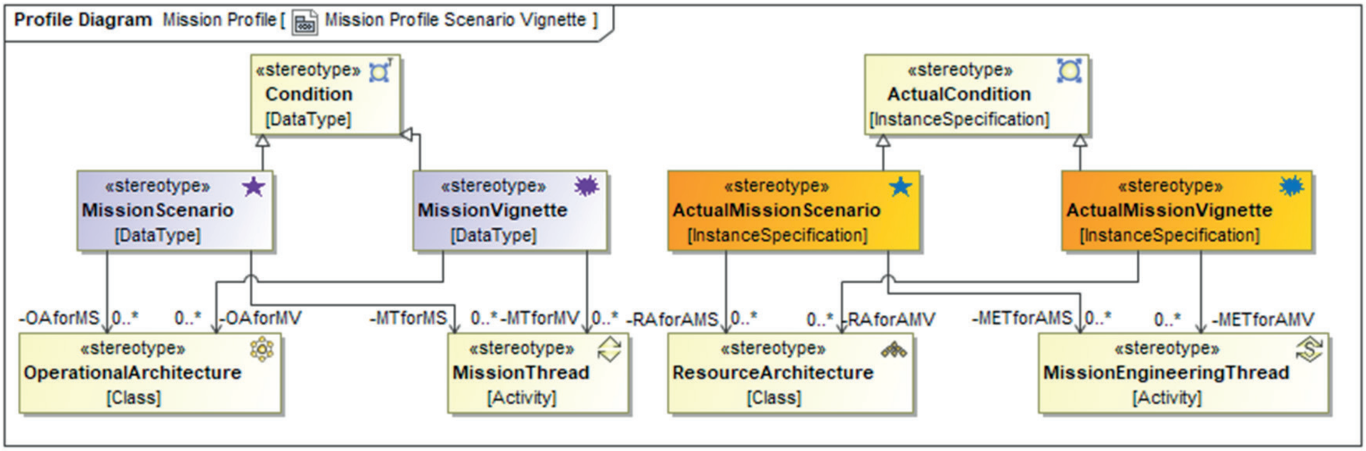

Profile Diagram Mission Profile [Mission Profile Scenario Vignette]
«stereotype» Condition [DataType]
«stereotype» ActualCondition [InstanceSpecification]
«stereotype» MissionScenario [DataType]
«stereotype» MissionVignette [DataType]
«stereotype» ActualMissionScenario [InstanceSpecification]
«stereotype» ActualMissionVignette [InstanceSpecification]
-OAforMS 0..*
0..* -OAforMV
-MTforMS
0..* -MTforMV
0..* -RAforAMS
0..*
0..* -RAforAMV
-METforAMS 0..*
0..* -METforAMV
«stereotype» OperationalArchitecture [Class]
«stereotype» MissionThread [Activity]
«stereotype» ResourceArchitecture [Class]
«stereotype» MissionEngineeringThread [Activity]

Figure 6. Scenario and vignette extensions to UAF.

Since both concepts describe a set of information relative to a mission, the most useful stereotype to extend is a Condition, with each of them having their own elements that relate to the contextual information described in the MEG. Modelers can then create Actual Conditions that have specific values for the appropriate Scenario and Vignette, as well as the thresholds for determining success, and then apply them to the specific Actual Missions within their model, providing the necessary traceability to their missions. Since Scenario and Vignette can be applicable to any Mission Type (as seen in Figure 5), the Scenario should get applied to the top-level Actual Mission in the model, and Vignettes should get created and applied to each Actual Mission below the top-level one.

In the Hoth example, Figure 7 shows how the Scenario and Vignette elements get defined and applied to the appropriate missions. On the left are a default Mission Scenario and Vignette. These elements will be included in the profile as examples in the same way as DLOD and DOTMLPF projects are. These have been extended for the Hoth Battle for a Planetary Invasion scenario, Ground Attack vignette, and Air Attack vignette. These can include additional conditions and values. Along the bottom are a set of conditions that can be used throughout the model regarding the environment, topography, and political situation. These are used by the instances of scenario and vignette on the right. These are then linked to the Mission definitions so that the Mission actuals can reference the Vignette and Scenario actuals. In this example, the actual scenario contains the vignettes.

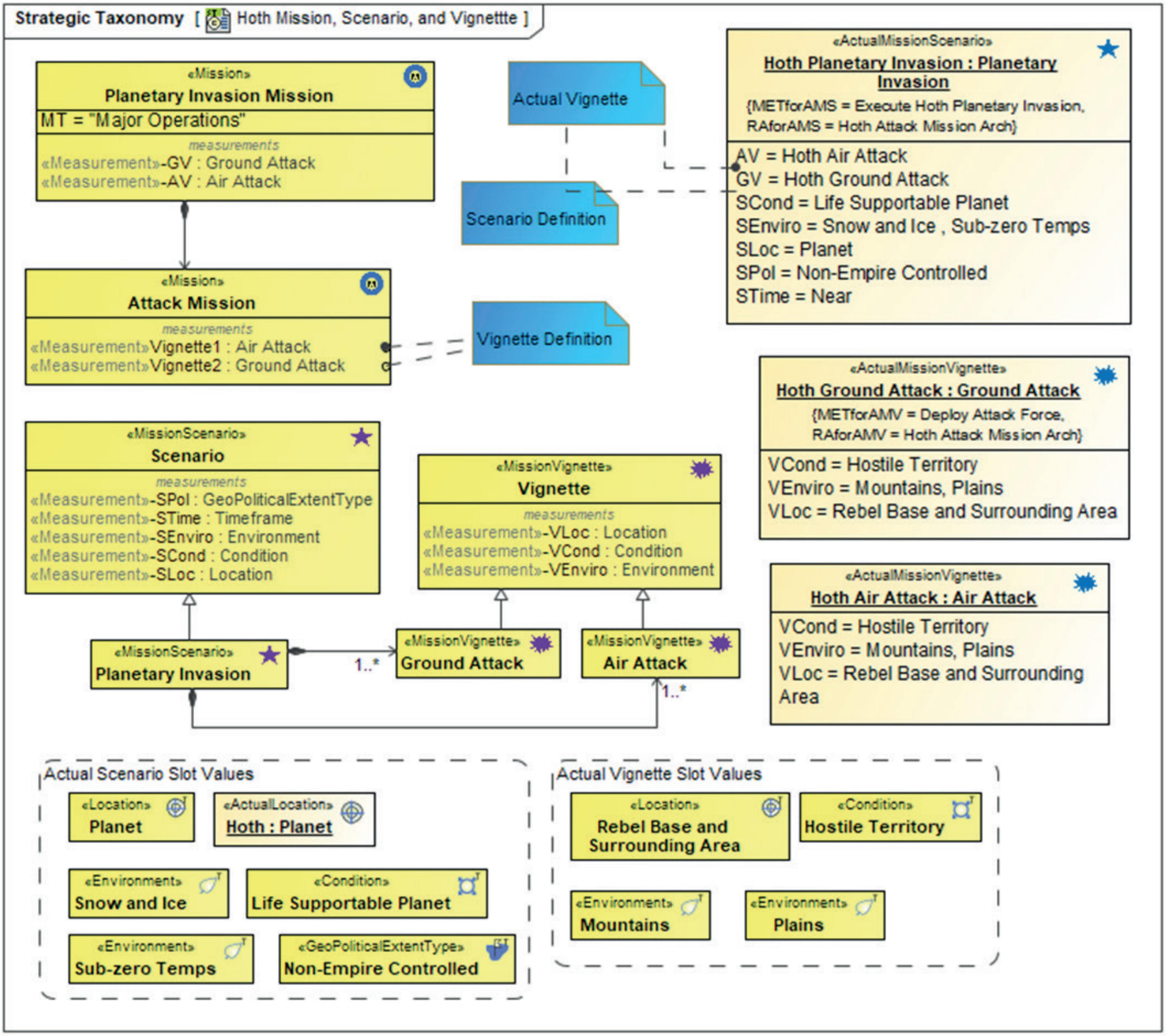

Strategic Taxonomy [Hoth Mission, Scenario, and Vignettte]
«Mission»
Planetary Invasion Mission
MT = "Major Operations"
measurements
«Measurement»-GV : Ground Attack
«Measurement»-AV : Air Attack
Actual Vignette
Scenario Definition
«ActualMissionScenario»
Hoth Planetary Invasion : Planetary Invasion
{METforAMS = Execute Hoth Planetary Invasion,
RAforAMS = Hoth Attack Mission Arch}
AV = Hoth Air Attack
GV = Hoth Ground Attack
SCond = Life Supportable Planet
SEnviro = Snow and Ice , Sub-zero Temps
SLoc = Planet
SPol = Non-Empire Controlled
STime = Near
«Mission»
Attack Mission
measurements
«Measurement»Vignette1 : Air Attack
«Measurement»Vignette2 : Ground Attack
Vignette Definition
«ActualMissionVignette»
Hoth Ground Attack : Ground Attack
{METforAMV = Deploy Attack Force,
RAforAMV = Hoth Attack Mission Arch}
VCond = Hostile Territory
VEnviro = Mountains, Plains
VLoc = Rebel Base and Surrounding Area
«MissionScenario»
Scenario
measurements
«Measurement»-SPol : GeoPoliticalExtentType
«Measurement»-STime : Timeframe
«Measurement»-SEnviro : Environment
«Measurement»-SCond : Condition
«Measurement»-SLoc : Location
«MissionVignette»
Vignette
measurements
«Measurement»-VLoc : Location
«Measurement»-VCond : Condition
«Measurement»-VEnviro : Environment
«ActualMissionVignette»
Hoth Air Attack : Air Attack
VCond = Hostile Territory
VEnviro = Mountains, Plains
VLoc = Rebel Base and Surrounding Area
«MissionScenario»
Planetary Invasion
1..*
«MissionVignette»
Ground Attack
«MissionVignette»
Air Attack
1..*
Actual Scenario Slot Values
«Location»
Planet
«ActualLocation»
Hoth : Planet
«Environment»
Snow and Ice
«Condition»
Life Supportable Planet
«Environment»
Sub-zero Temps
«GeoPoliticalExtentType»
Non-Empire Controlled
Actual Vignette Slot Values
«Location»
Rebel Base and Surrounding Area
«Condition»
Hostile Territory
«Environment»
Mountains
«Environment»
Plains

Figure 7. Definition of scenario and vignette.

2.4. Mission relationships

Two relationships are used to connect mission types to MTs (Process Defines Initiative), shown in Figure 8, and actual missions to METs (Process Adapts to Initiative), shown in Figure 9. Figure 9 shows the structure of the Hoth Invasion, which is an instance of the Planetary Invasion Mission defined in Figure 4. This Actual Mission is made up of the Planning and Execution Phase as well as the Landing Mission, Attack Mission, and Scout mission. These Missions each have Planning and Execution Phases. The Execution phases all have METs mapped to them. The Hoth AMEP Execution Phase has defined goals as well as Operational and Resource Architecture.

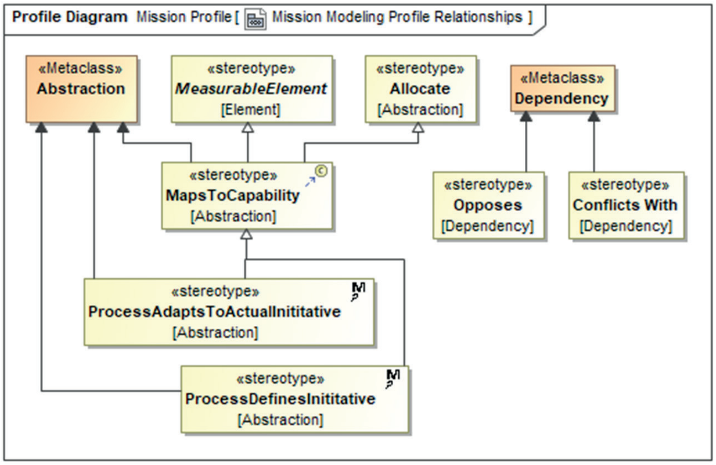

Profile Diagram Mission Profile [Mission Modeling Profile Relationships]
«Metaclass» Abstraction
«stereotype» MeasurableElement [Element]
«stereotype» Allocate [Abstraction]
«Metaclass» Dependency
«stereotype» MapsToCapability [Abstraction]
«stereotype» Opposes [Dependency]
«stereotype» Conflicts With [Dependency]
«stereotype» ProcessAdaptsToActualInititative [Abstraction]
«stereotype» ProcessDefinesInititative [Abstraction]

Figure 8. Mission relationships.

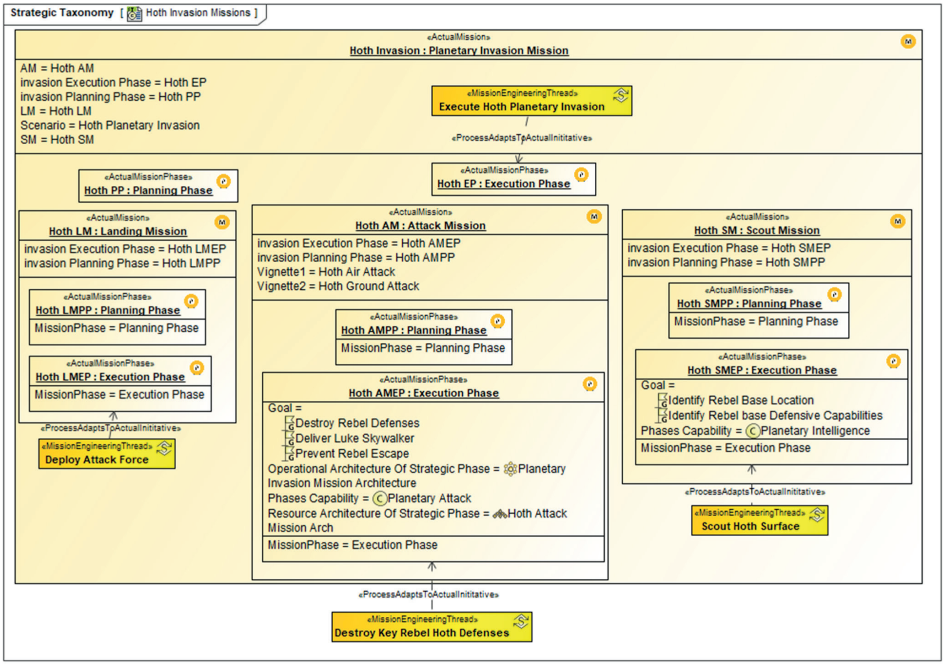

Strategic Taxonomy [Hoth Invasion Missions]
«ActualMission» Hoth Invasion : Planetary Invasion Mission
AM = Hoth AM
invasion Execution Phase = Hoth EP
invasion Planning Phase = Hoth PP
LM = Hoth LM
Scenario = Hoth Planetary Invasion
SM = Hoth SM
«MissionEngineeringThread» Execute Hoth Planetary Invasion
«ProcessAdaptsToActualInititative»
«ActualMissionPhase» Hoth PP : Planning Phase
«ActualMissionPhase» Hoth EP : Execution Phase
«ActualMission» Hoth LM : Landing Mission
invasion Execution Phase = Hoth LMEP
invasion Planning Phase = Hoth LMPP
«ActualMissionPhase» Hoth LMPP : Planning Phase
MissionPhase = Planning Phase
«ActualMissionPhase» Hoth LMEP : Execution Phase
MissionPhase = Execution Phase
«ProcessAdaptsToActualInititative»
«MissionEngineeringThread» Deploy Attack Force
«ActualMission» Hoth AM : Attack Mission
invasion Execution Phase = Hoth AMEP
invasion Planning Phase = Hoth AMPP
Vignette1 = Hoth Air Attack
Vignette2 = Hoth Ground Attack
«ActualMissionPhase» Hoth AMPP : Planning Phase
MissionPhase = Planning Phase
«ActualMissionPhase» Hoth AMEP : Execution Phase
Goal =
Destroy Rebel Defenses
Deliver Luke Skywalker
Prevent Rebel Escape
Operational Architecture Of Strategic Phase = Planetary Invasion Mission Architecture
Phases Capability = Planetary Attack
Resource Architecture Of Strategic Phase = Hoth Attack Mission Arch
MissionPhase = Execution Phase
«ActualMission» Hoth SM : Scout Mission
invasion Execution Phase = Hoth SMEP
invasion Planning Phase = Hoth SMPP
«ActualMissionPhase» Hoth SMPP : Planning Phase
MissionPhase = Planning Phase
«ActualMissionPhase» Hoth SMEP : Execution Phase
Goal =
Identify Rebel Base Location
Identify Rebel base Defensive Capabilities
Phases Capability = Planetary Intelligence
MissionPhase = Execution Phase
«ProcessAdaptsToActualInititative»
«MissionEngineeringThread» Scout Hoth Surface
«ProcessAdaptsToActualInititative»
«MissionEngineeringThread» Destroy Key Rebel Hoth Defenses

Figure 9. Actual missions and mission phases.

2.4.1. Conflicting elements

Opposition and Conflict are inherent in ME. Some of these are obvious in the context of this mission: the Empire Forces attack the Rebel Forces, Energy Cannons attack the Defense Shield, etc. Others are not so obvious, such as the conflict within the Goals of the Mission shown in Figure 10. The Goal to Capture Luke Skywalker reduces the chances of Destroy Rebel Defenses and Prevent Rebel Escape. Normally, the Empire executes its missions with extreme prejudice, preferring to destroy a planet rather than allowing enemies to escape or information to be released. Since they had to attack with conventional forces and to do so with great care, they were unable to destroy the forces or prevent the Rebel escape. Highlighting these conflicting elements would help to ensure a successful outcome and provide a means of mitigating risk and other aspects. Each goal is further decomposed into its objectives. Objectives define short term accomplishments while goals are long term.

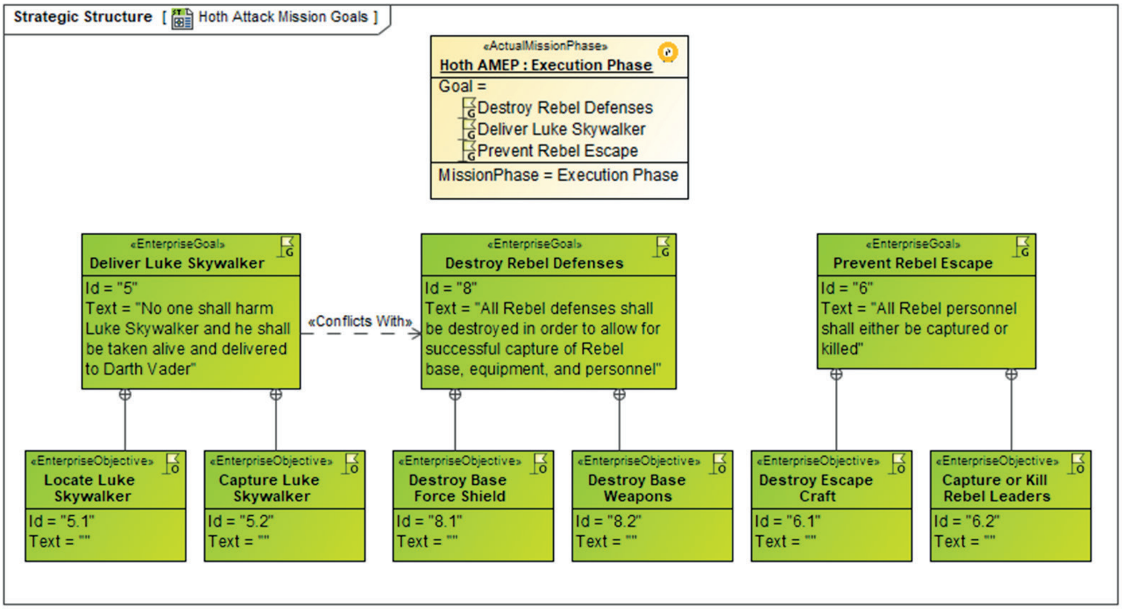


Figure 10. Mission goals and objectives.

2.4.2. Linking strategy to execution

The goals, drivers, challenges, opportunities, mission phases, capabilities, and systems are linked together in Figure 11. The Hoth AMEP execution phase phases the goals of Destroy Rebel Defenses, prevent Rebel Escape, and Deliver Luke Skywalker and the Planetary Attack capability. This means that they are realized during this phase. The Resource and Operational Architectures implement the mission phase and the MET is executed. Risks of the Loss of Empire Forces and Rebel Forces Escape are identified for the opportunities. Mitigation strategies can be developed for these risks.

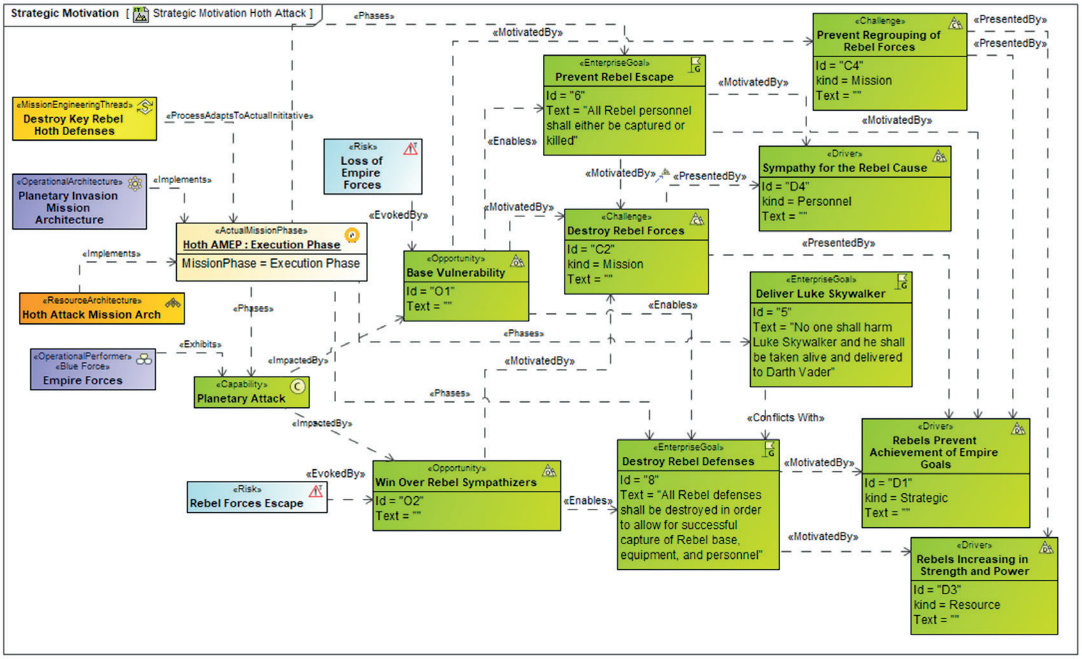


Figure 11. Mission drivers, goals, challenges, opportunities and capabilities.

3. DEVELOPING AN OPERATIONAL ARCHITECTURE

3.1. The Blue Force operational architecture

The Empire's operational architecture, shown in Figure 12, lays out at a logical level the main elements of the Empire forces that are needed to execute a Planetary Invasion. Note that the Empire must also estimate what Rebel Forces may be present, so that they can account for their interactions with the Empire's forces.

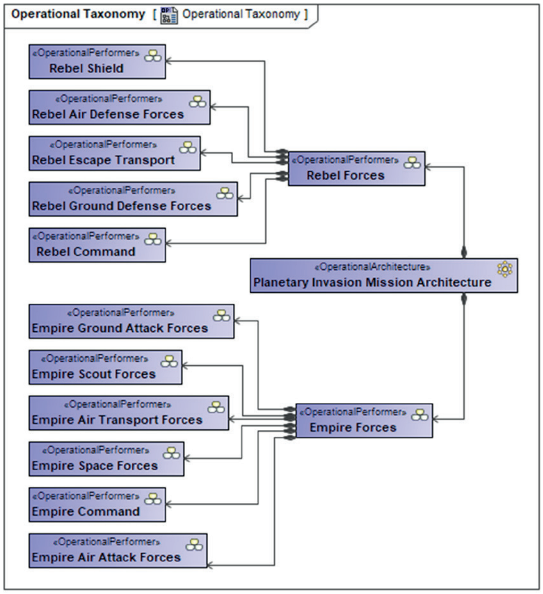
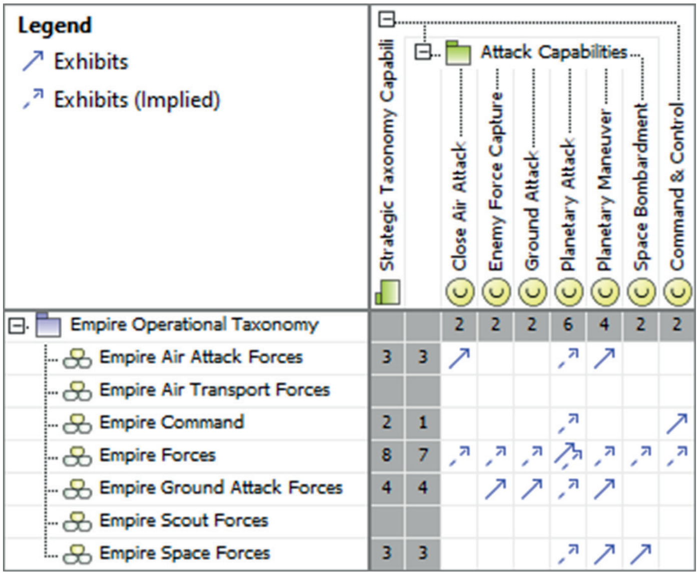

Operational Taxonomy [Operational Taxonomy]
«OperationalPerformer» Rebel Shield
«OperationalPerformer» Rebel Air Defense Forces
«OperationalPerformer» Rebel Escape Transport
«OperationalPerformer» Rebel Ground Defense Forces
«OperationalPerformer» Rebel Command
«OperationalPerformer» Rebel Forces
«OperationalArchitecture» Planetary Invasion Mission Architecture
«OperationalPerformer» Empire Ground Attack Forces
«OperationalPerformer» Empire Scout Forces
«OperationalPerformer» Empire Air Transport Forces
«OperationalPerformer» Empire Space Forces
«OperationalPerformer» Empire Command
«OperationalPerformer» Empire Air Attack Forces
«OperationalPerformer» Empire Forces

Figure 12. Attack phase operational taxonomy

These candidate logical elements are mapped to the required capabilities to ensure that they have all been addressed as shown in Figure 13.



Legend
↗ Exhibits
⇢ Exhibits (Implied)

	Strategic Taxonomy Capabili	Attack Capabilities...	Close Air Attack	Enemy Force Capture	Ground Attack	Planetary Attack	Planetary Maneuver	Space Bombardment	Command & Control
Empire Operational Taxonomy			2	2	2	6	4	2	2
Empire Air Attack Forces	3	3	↗			⇢	↗		
Empire Air Transport Forces									
Empire Command	2	1				⇢			↗
Empire Forces	8	7	⇢	⇢	⇢	↗⇢	⇢	⇢	⇢
Empire Ground Attack Forces	4	4		↗	↗	⇢	↗		
Empire Scout Forces									
Empire Space Forces	3	3				⇢	↗	↗	

Figure 13. Operational performers to capabilities mapping table.

Note that Empire Air Transport and Empire Scout Forces are not included as they are not required for this phase. Having defined these structural elements, their interactions can be defined using the internal connectivity diagram, similar to a SysML IBD as shown below in Figure 14.

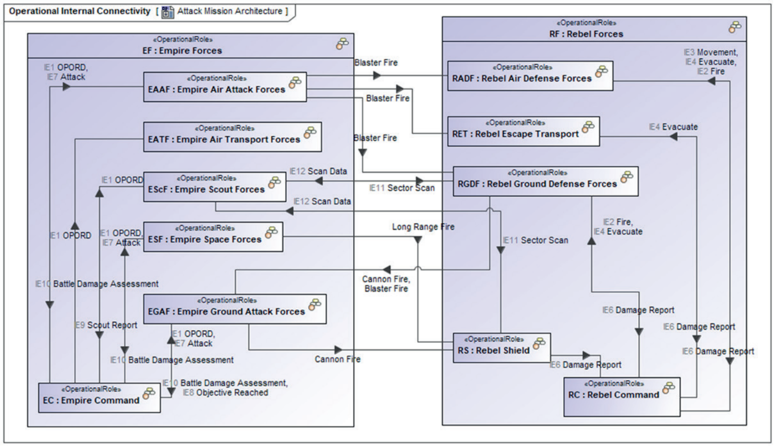

Operational Internal Connectivity [Attack Mission Architecture]
«OperationalRole» EF : Empire Forces
«OperationalRole» EAAF : Empire Air Attack Forces
«OperationalRole» EATF : Empire Air Transport Forces
«OperationalRole» EScF : Empire Scout Forces
«OperationalRole» ESF : Empire Space Forces
«OperationalRole» EGAF : Empire Ground Attack Forces
«OperationalRole» EC : Empire Command
«OperationalRole» RF : Rebel Forces
«OperationalRole» RADF : Rebel Air Defense Forces
«OperationalRole» RET : Rebel Escape Transport
«OperationalRole» RGDF : Rebel Ground Defense Forces
«OperationalRole» RS : Rebel Shield
«OperationalRole» RC : Rebel Command
IE1 OPORD, IE7 Attack
IE1 OPORD
IE10 Battle Damage Assessment
IE9 Scout Report
IE10 Battle Damage Assessment, IE8 Objective Reached
IE12 Scan Data
IE11 Sector Scan
Blaster Fire
Long Range Fire
Cannon Fire, Blaster Fire
Cannon Fire
IE3 Movement, IE4 Evacuate, IE2 Fire
IE4 Evacuate
IE2 Fire, IE4 Evacuate
IE6 Damage Report

Figure 14. Attack mission architecture.

The mission architecture shows internal communication links as well as weapons fire and scan data between the opposing forces. The Rebel Forces are those which were identified during the Scout Mission. Showing these interactions ensures that the required firepower and tactical resources are available for the mission.

3.2. Operational activities

The Empire identifies applicable operational activities and that each element in the Operational Architecture could perform, in the context of a Scout, Landing, or Attack mission in support of a Planetary Invasion. These are mapped to one another and correspond to the defined behavior for these elements in Figure 15. The rebel and empire elements are modeled on separate diagrams.

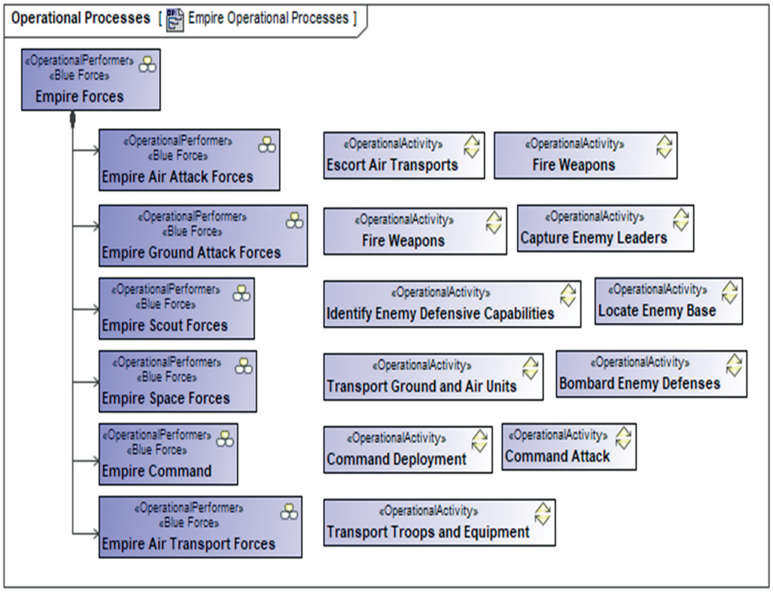

Operational Processes [Empire Operational Processes]
«OperationalPerformer» «Blue Force» Empire Forces
«OperationalPerformer» «Blue Force» Empire Air Attack Forces
«OperationalPerformer» «Blue Force» Empire Ground Attack Forces
«OperationalPerformer» «Blue Force» Empire Scout Forces
«OperationalPerformer» «Blue Force» Empire Space Forces
«OperationalPerformer» «Blue Force» Empire Command
«OperationalPerformer» «Blue Force» Empire Air Transport Forces
«OperationalActivity» Escort Air Transports
«OperationalActivity» Fire Weapons
«OperationalActivity» Fire Weapons
«OperationalActivity» Capture Enemy Leaders
«OperationalActivity» Identify Enemy Defensive Capabilities
«OperationalActivity» Locate Enemy Base
«OperationalActivity» Transport Ground and Air Units
«OperationalActivity» Bombard Enemy Defenses
«OperationalActivity» Command Deployment
«OperationalActivity» Command Attack
«OperationalActivity» Transport Troops and Equipment

Figure 15. Empire forces and activities.

These define the activities that can be performed by the Empire forces. These will be used during the execution of the mission. Figure 16 defines the functionality defined for the Rebel Forces. This set of defensive activities will be evaluated against the Empire offensive activities.

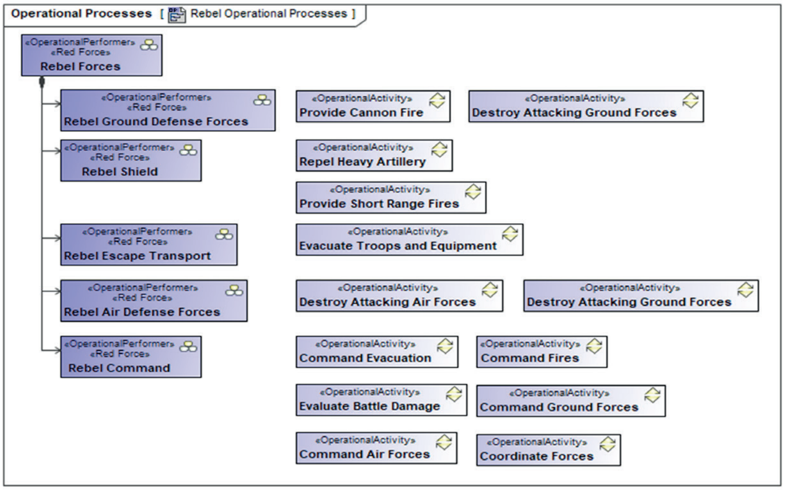

Operational Processes [Rebel Operational Processes]
«OperationalPerformer» «Red Force» Rebel Forces
«OperationalPerformer» «Red Force» Rebel Ground Defense Forces
«OperationalPerformer» «Red Force» Rebel Shield
«OperationalPerformer» «Red Force» Rebel Escape Transport
«OperationalPerformer» «Red Force» Rebel Air Defense Forces
«OperationalPerformer» «Red Force» Rebel Command
«OperationalActivity» Provide Cannon Fire
«OperationalActivity» Destroy Attacking Ground Forces
«OperationalActivity» Repel Heavy Artillery
«OperationalActivity» Provide Short Range Fires
«OperationalActivity» Evacuate Troops and Equipment
«OperationalActivity» Destroy Attacking Air Forces
«OperationalActivity» Destroy Attacking Ground Forces
«OperationalActivity» Command Evacuation
«OperationalActivity» Command Fires
«OperationalActivity» Evaluate Battle Damage
«OperationalActivity» Command Ground Forces
«OperationalActivity» Command Air Forces
«OperationalActivity» Coordinate Forces

Figure 16. Rebel performers and activities.

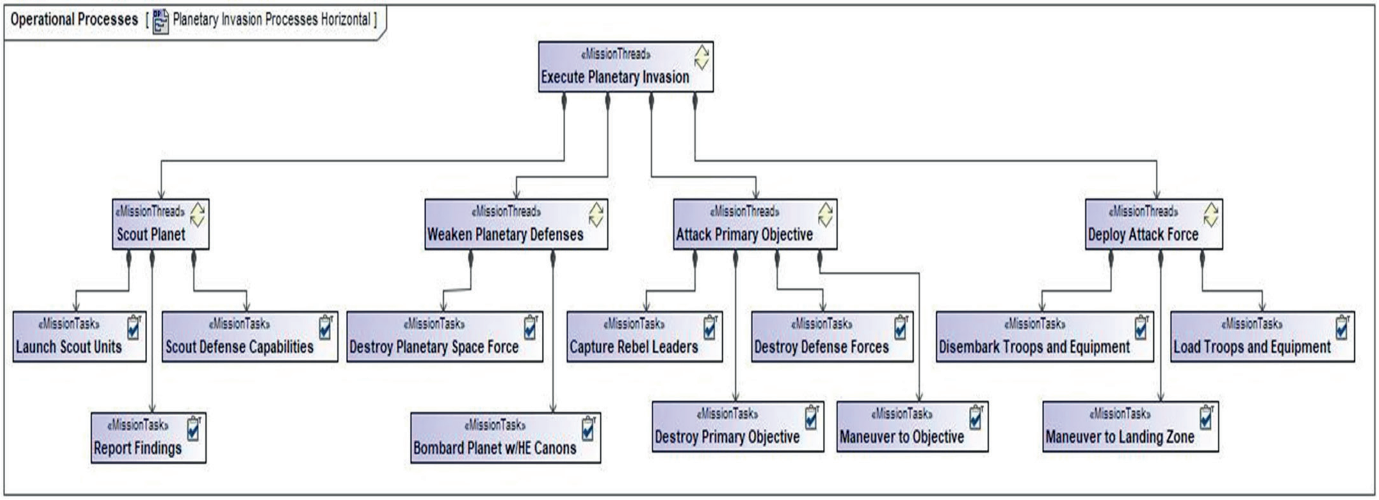


Figure 17. Planetary invasion mission threads and tasks.

3.3. Mission thread definition

The Empire's doctrine lays out the Mission Threads and Mission Tasks for each Mission Phase. Figure 17 shows the breakdown of Mission Threads and Mission Tasks for the Execution Phase of a Planetary Invasion. It is broken down into Mission Threads of Scout Planet, Weaken Planetary Defenses, Attack Primary Objective and Deploy Attack Force. Each of these are further decomposed into Mission Tasks.

For each Mission Thread and Mission Task an Operational Activity Diagram is developed to describe what Operational Activities are needed to accomplish each Mission Thread, and what part of the Operational Architecture will be expected to perform them. These were defined in Figure 15. Libraries of these should be built up over time to minimize the required time for mission engineering and ensure correctness and compliance with standards and doctrine. Figure 18 shows the description of the "Destroy Defense Forces" operational process diagram.

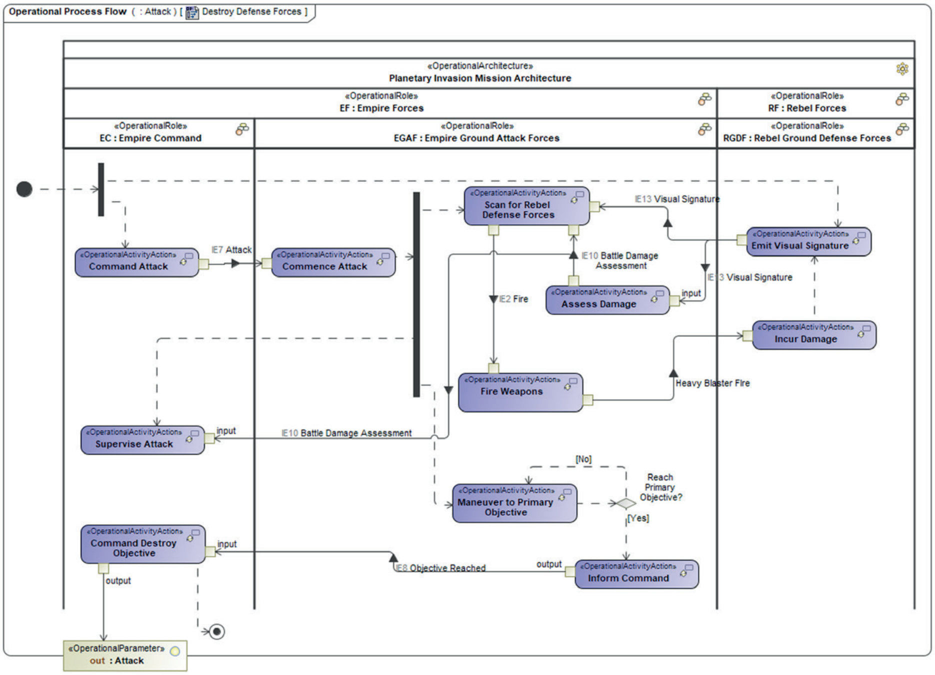
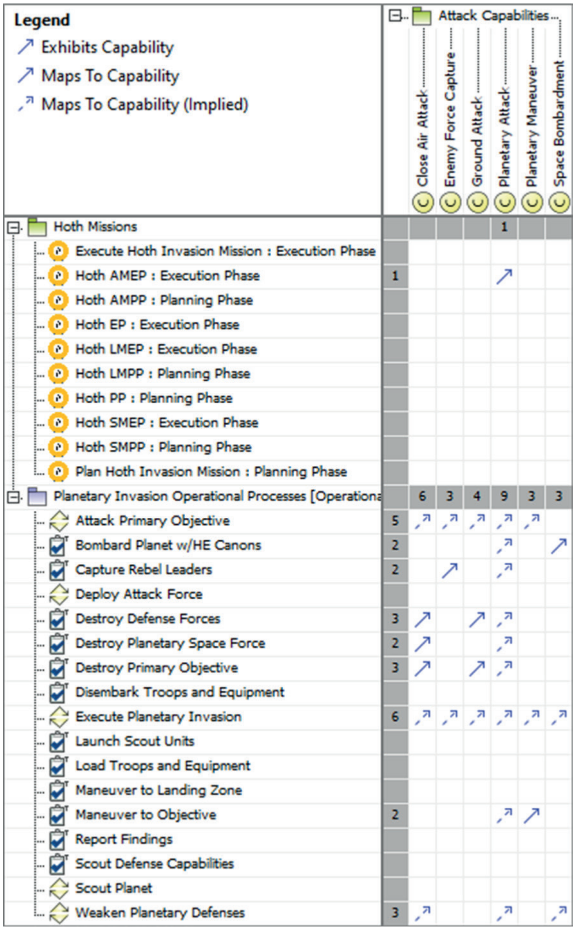

Operational Process Flow (: Attack) [Destroy Defense Forces]
«OperationalArchitecture»
Planetary Invasion Mission Architecture
«OperationalRole»
EF : Empire Forces
«OperationalRole»
RF : Rebel Forces
«OperationalRole»
EC : Empire Command
«OperationalRole»
EGAF : Empire Ground Attack Forces
«OperationalRole»
RGDF : Rebel Ground Defense Forces
«OperationalActivityAction»
Command Attack
IE7 Attack
«OperationalActivityAction»
Commence Attack
«OperationalActivityAction»
Scan for Rebel Defense Forces
IE13 Visual Signature
«OperationalActivityAction»
Emit Visual Signature
IE10 Battle Damage Assessment
IE13 Visual Signature
IE2 Fire
«OperationalActivityAction»
Assess Damage
input
«OperationalActivityAction»
Incur Damage
«OperationalActivityAction»
Fire Weapons
Heavy Blaster Fire
«OperationalActivityAction»
Supervise Attack
input
IE10 Battle Damage Assessment
[No]
Reach Primary Objective?
«OperationalActivityAction»
Maneuver to Primary Objective
[Yes]
«OperationalActivityAction»
Command Destroy Objective
input
IE8 Objective Reached
output
«OperationalActivityAction»
Inform Command
output
«OperationalParameter»
out : Attack

Figure 18. Destroy defense forces.

Figure 19 shows the Strategic Traceability between Mission Threads, Tasks, and Actual Missions. This demonstrates that the capabilities are addressed by the functional elements. Only the Attack Mission is shown for simplicity.

3.4. Differentiation between Enemy/Friendly/Neutral

ME models require the identification of different forces such as enemy, friendly, neutral, or others. This can correspond to individual elements as well as organizations and groups. The most useful way of accomplishing this is through a set of stereotypes that allow tracking these elements easily within the model, as well as allowing for unique formatting (e.g., colors) that clearly identify them in diagrams. The ME Profile adds 5 of these Force Designation stereotypes, with an overarching stereotype that they specialize, as seen in Figure 20. This also allows modelers to add additional stereotypes by simply inheriting from the overarching Force Designation stereotype. The term "Force Designation" was chosen as the term "Force Type" implies Army, Navy, Air Force, etc., and could be confusing. These types could be added by an engineer to extend the profile.



Legend
↗ Exhibits Capability
↗ Maps To Capability
⇗ Maps To Capability (Implied)

Legend		Attack Capabilities...					
		Close Air Attack	Enemy Force Capture	Ground Attack	Planetary Attack	Planetary Maneuver	Space Bombardment
Hoth Missions					1		
Execute Hoth Invasion Mission : Execution Phase							
Hoth AMEP : Execution Phase	1				↗		
Hoth AMPP : Planning Phase							
Hoth EP : Execution Phase							
Hoth LMEP : Execution Phase							
Hoth LMPP : Planning Phase							
Hoth PP : Planning Phase							
Hoth SMEP : Execution Phase							
Hoth SMPP : Planning Phase							
Plan Hoth Invasion Mission : Planning Phase							
Planetary Invasion Operational Processes [Operationa		6	3	4	9	3	3
Attack Primary Objective	5	⇗	⇗	⇗	⇗	⇗	
Bombard Planet w/HE Canons	2				⇗		↗
Capture Rebel Leaders	2		↗		⇗		
Deploy Attack Force							
Destroy Defense Forces	3	↗		↗	⇗		
Destroy Planetary Space Force	2	↗			⇗		
Destroy Primary Objective	3	↗		↗	⇗		
Disembark Troops and Equipment							
Execute Planetary Invasion	6	⇗	⇗	⇗	⇗	⇗	⇗
Launch Scout Units							
Load Troops and Equipment							
Maneuver to Landing Zone							
Maneuver to Objective	2				⇗	↗	
Report Findings							
Scout Defense Capabilities							
Scout Planet							
Weaken Planetary Defenses	3	⇗			⇗		⇗

Figure 19. Strategic traceability between mission threads, tasks, and actual missions.

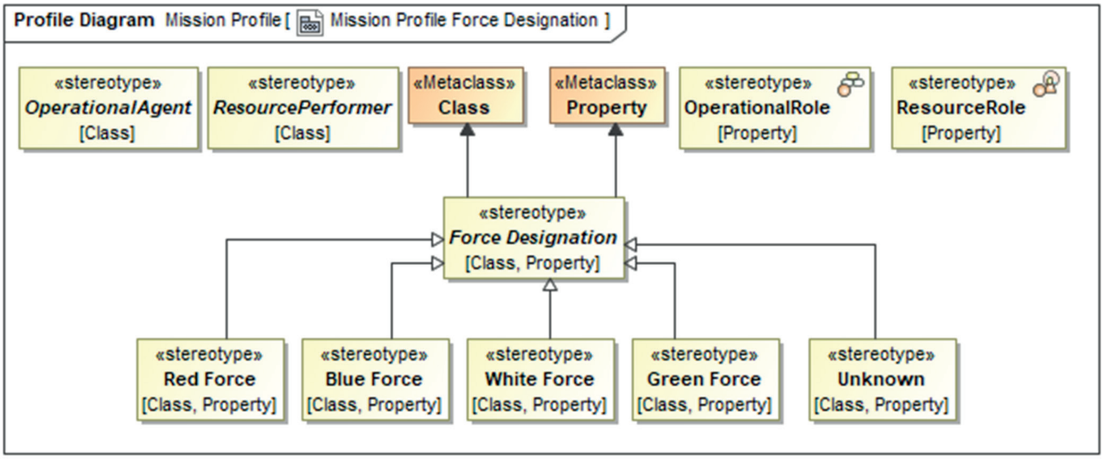

Profile Diagram Mission Profile [Mission Profile Force Designation]
«stereotype» OperationalAgent [Class]
«stereotype» ResourcePerformer [Class]
«Metaclass» Class
«Metaclass» Property
«stereotype» OperationalRole [Property]
«stereotype» ResourceRole [Property]
«stereotype» Force Designation [Class, Property]
«stereotype» Red Force [Class, Property]
«stereotype» Blue Force [Class, Property]
«stereotype» White Force [Class, Property]
«stereotype» Green Force [Class, Property]
«stereotype» Unknown [Class, Property]

Figure 20. Force designation definition profile diagram.

Figure 21 shows the opposing Empire and Rebel forces. The Rebel forces are shown at the top in red. Empire forces are shown in blue. The force designations can be applied to either the definition as shown here or to the role elements in an internal connectivity diagram. Other force designations may include civilians, commercial operations, allies, etc. The opposes relationship originally defined in Figure 8 shows mission elements that will contend/attack/fight one another.

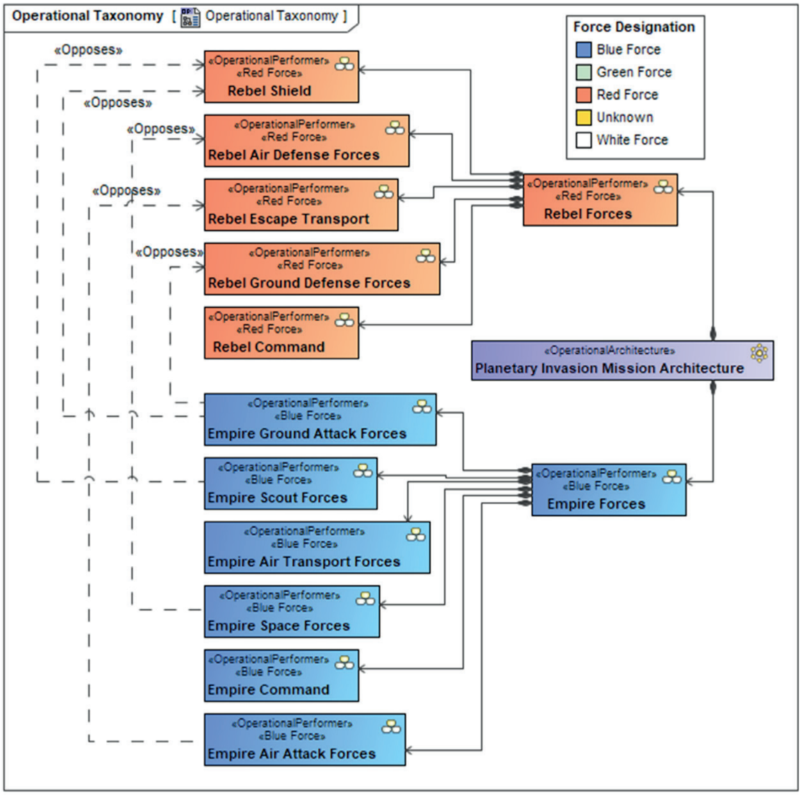

Operational Taxonomy [Operational Taxonomy]
Force Designation
Blue Force
Green Force
Red Force
Unknown
White Force
«Opposes»
«Opposes»
«Opposes»
«Opposes»
«Opposes»
«OperationalPerformer» «Red Force» Rebel Shield
«OperationalPerformer» «Red Force» Rebel Air Defense Forces
«OperationalPerformer» «Red Force» Rebel Escape Transport
«OperationalPerformer» «Red Force» Rebel Ground Defense Forces
«OperationalPerformer» «Red Force» Rebel Command
«OperationalPerformer» «Red Force» Rebel Forces
«OperationalArchitecture» Planetary Invasion Mission Architecture
«OperationalPerformer» «Blue Force» Empire Ground Attack Forces
«OperationalPerformer» «Blue Force» Empire Scout Forces
«OperationalPerformer» «Blue Force» Empire Air Transport Forces
«OperationalPerformer» «Blue Force» Empire Space Forces
«OperationalPerformer» «Blue Force» Empire Command
«OperationalPerformer» «Blue Force» Empire Air Attack Forces
«OperationalPerformer» «Blue Force» Empire Forces

Figure 21. Red and blue mission performer elements.

3.5. Goals, objectives, effects, and outcomes

As mentioned previously, goals and objectives are modeled as types of requirements. As such, they come with unique identifiers, can be nested, and can make use of all the relationships afforded to requirements. Figure 10 listed the goals and objectives of the attack mission. A portion of the Execute Planetary Invasion MT is shown in Figure 22. The different mission tasks satisfy the outcomes and objectives of the mission. In this way, it demonstrates that a ME solution should also be able to achieve the goals and objectives.

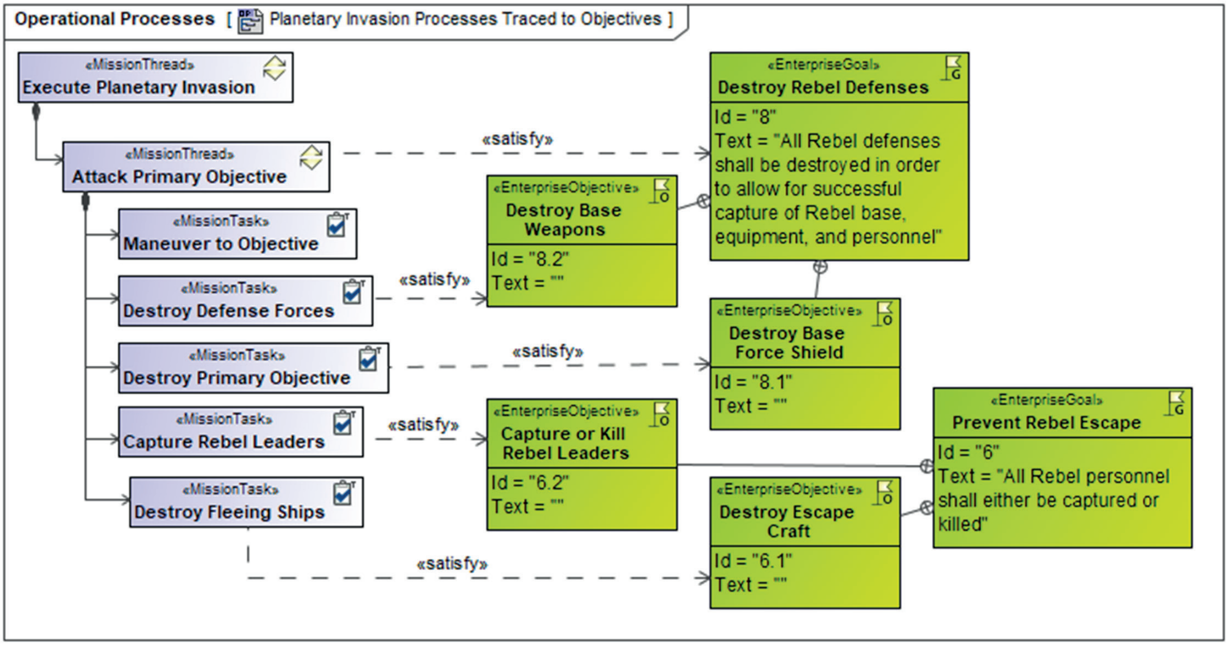
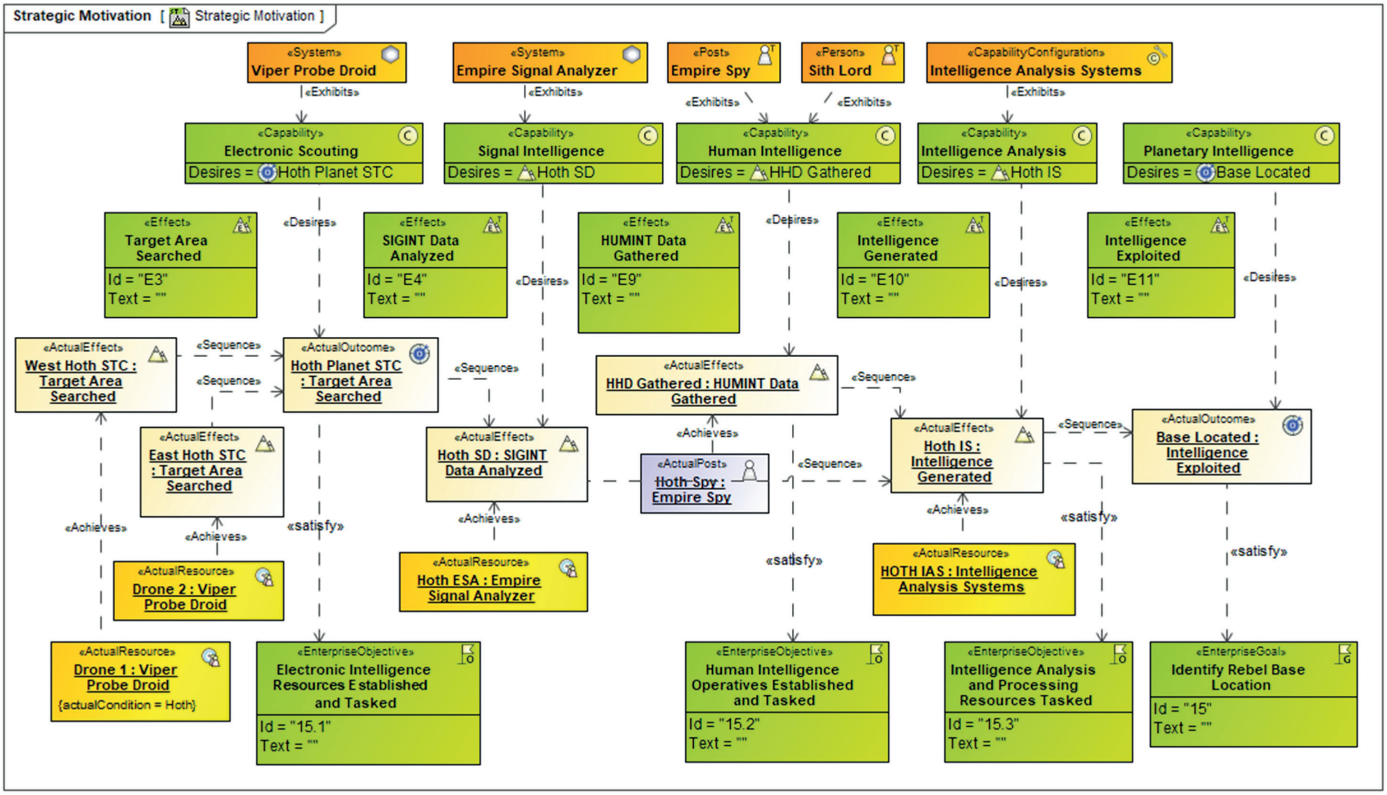

Operational Processes [Planetary Invasion Processes Traced to Objectives]
«MissionThread» Execute Planetary Invasion
«MissionThread» Attack Primary Objective
«MissionTask» Maneuver to Objective
«MissionTask» Destroy Defense Forces
«MissionTask» Destroy Primary Objective
«MissionTask» Capture Rebel Leaders
«MissionTask» Destroy Fleeing Ships
«satisfy»
«satisfy»
«satisfy»
«satisfy»
«satisfy»
«EnterpriseGoal» Destroy Rebel Defenses
Id = "8"
Text = "All Rebel defenses shall be destroyed in order to allow for successful capture of Rebel base, equipment, and personnel"
«EnterpriseObjective» Destroy Base Weapons
Id = "8.2"
Text = ""
«EnterpriseObjective» Destroy Base Force Shield
Id = "8.1"
Text = ""
«EnterpriseObjective» Capture or Kill Rebel Leaders
Id = "6.2"
Text = ""
«EnterpriseObjective» Destroy Escape Craft
Id = "6.1"
Text = ""
«EnterpriseGoal» Prevent Rebel Escape
Id = "6"
Text = "All Rebel personnel shall either be captured or killed"

Figure 22. Mission threads and tasks linked to goals and objectives.

Figure 23 shows the objectives of the intelligence gathering mission as well as the effects, outcomes and the systems that will achieve those effects and outcomes. The effects and outcomes are sequences corresponding to deploying drones and spies, gathering information, sensing signals, analyzing those signals and finally synthesizing that intelligence to show the location of the rebel base.



Strategic Motivation [Strategic Motivation]
«System» Viper Probe Droid
«System» Empire Signal Analyzer
«Post» Empire Spy
«Person» Sith Lord
«CapabilityConfiguration» Intelligence Analysis Systems
«Exhibits»
«Exhibits»
«Exhibits»
«Exhibits»
«Exhibits»
«Capability» Electronic Scouting
Desires = Hoth Planet STC
«Capability» Signal Intelligence
Desires = Hoth SD
«Capability» Human Intelligence
Desires = HHD Gathered
«Capability» Intelligence Analysis
Desires = Hoth IS
«Capability» Planetary Intelligence
Desires = Base Located
«Effect» Target Area Searched
Id = "E3"
Text = ""
«Desires»
«Effect» SIGINT Data Analyzed
Id = "E4"
Text = ""
«Desires»
«Effect» HUMINT Data Gathered
Id = "E9"
Text = ""
«Desires»
«Effect» Intelligence Generated
Id = "E10"
Text = ""
«Desires»
«Effect» Intelligence Exploited
Id = "E11"
Text = ""
«Desires»
«ActualEffect» West Hoth STC : Target Area Searched
«Sequence»
«Sequence»
«ActualOutcome» Hoth Planet STC : Target Area Searched
«Sequence»
«ActualEffect» HHD Gathered : HUMINT Data Gathered
«Sequence»
«ActualEffect» East Hoth STC : Target Area Searched
«ActualEffect» Hoth SD : SIGINT Data Analyzed
«Achieves»
«ActualPost» Hoth Spy : Empire Spy
«Sequence»
«ActualEffect» Hoth IS : Intelligence Generated
«Sequence»
«ActualOutcome» Base Located : Intelligence Exploited
«Achieves»
«Achieves»
«satisfy»
«Achieves»
«Achieves»
«satisfy»
«satisfy»
«satisfy»
«ActualResource» Drone 2 : Viper Probe Droid
«ActualResource» Hoth ESA : Empire Signal Analyzer
«ActualResource» HOTH IAS : Intelligence Analysis Systems
«ActualResource» Drone 1 : Viper Probe Droid
{actualCondition = Hoth}
«EnterpriseObjective» Electronic Intelligence Resources Established and Tasked
Id = "15.1"
Text = ""
«EnterpriseObjective» Human Intelligence Operatives Established and Tasked
Id = "15.2"
Text = ""
«EnterpriseObjective» Intelligence Analysis and Processing Resources Tasked
Id = "15.3"
Text = ""
«EnterpriseGoal» Identify Rebel Base Location
Id = "15"
Text = ""

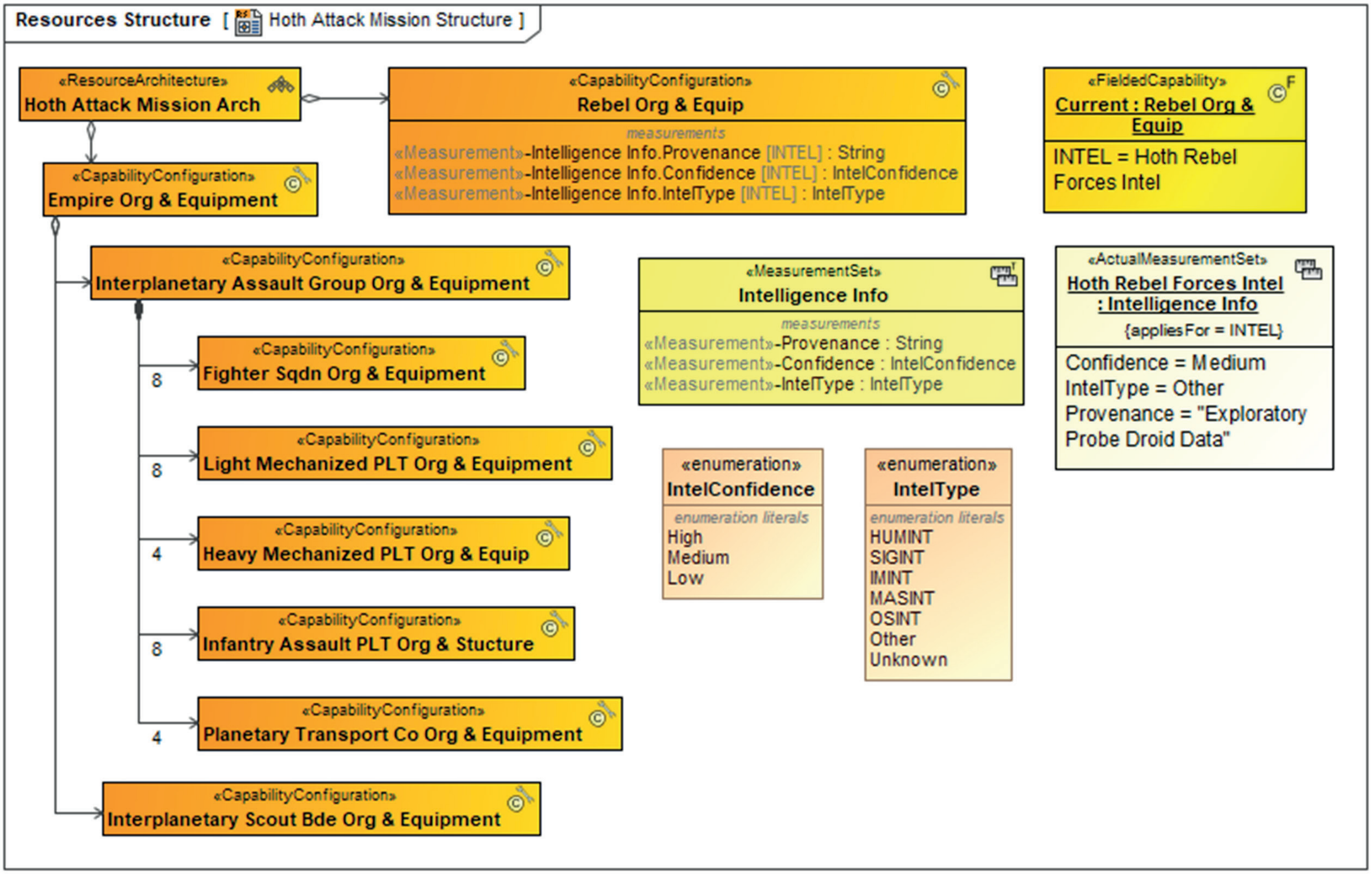
Figure 23. Mission objectives, effects, outcomes and achieving elements.

3.5.1. Provenance/Confidence of enemy resources

The rebel forces capabilities, forces, activities, strength, etc. have been discovered via the intelligence services. Two aspects of any intelligence are the provenance of the information and the level of confidence in the information as well as the source. The structure and behavioral elements created based on that intelligence should refer to the source (provenance) and corresponding confidence. Figure 24 shows a portion of the rebel forces. Information on the rebel forces has been collated by drones and spies. Enumerations have been defined for the profile providing Intel Confidence and Intel Type. Attributes correspond to the resource elements. These are instantiated as a fielded capability and specific values associated with them. In this case, the intelligence was gathered by an intelligence probe droid, the type is unknown, and the confidence level is medium.

3.5.2. Compliance/Conformance to doctrine/ standards.

The SysML requirement element has been extended to provide concepts of Ref Doctrine, Ref Publication, and Ref Standard (not shown for reasons of space). These provide the ability to link specific steps in an MT or MET, mission elements or the entire mission to atomic elements of doctrine. This can be crucial to ensure that proper procedures are followed when constructing missions. The UAF standard concept is also available but is typically at a more *"macro"* level of that deals with an entire document.



Resources Structure [Hoth Attack Mission Structure]
«ResourceArchitecture»
Hoth Attack Mission Arch
«CapabilityConfiguration»
Rebel Org & Equip
measurements
«Measurement»-Intelligence Info.Provenance [INTEL] : String
«Measurement»-Intelligence Info.Confidence [INTEL] : IntelConfidence
«Measurement»-Intelligence Info.IntelType [INTEL] : IntelType
«FieldedCapability»
Current : Rebel Org & Equip
INTEL = Hoth Rebel Forces Intel
«CapabilityConfiguration»
Empire Org & Equipment
«CapabilityConfiguration»
Interplanetary Assault Group Org & Equipment
«MeasurementSet»
Intelligence Info
measurements
«Measurement»-Provenance : String
«Measurement»-Confidence : IntelConfidence
«Measurement»-IntelType : IntelType
«ActualMeasurementSet»
Hoth Rebel Forces Intel : Intelligence Info
{appliesFor = INTEL}
Confidence = Medium
IntelType = Other
Provenance = "Exploratory Probe Droid Data"
«CapabilityConfiguration»
Fighter Sqdn Org & Equipment
8
«CapabilityConfiguration»
Light Mechanized PLT Org & Equipment
8
«CapabilityConfiguration»
Heavy Mechanized PLT Org & Equip
4
«CapabilityConfiguration»
Infantry Assault PLT Org & Stucture
8
«CapabilityConfiguration»
Planetary Transport Co Org & Equipment
4
«CapabilityConfiguration»
Interplanetary Scout Bde Org & Equipment
«enumeration»
IntelConfidence
enumeration literals
High
Medium
Low
«enumeration»
IntelType
enumeration literals
HUMINT
SIGINT
IMINT
MASINT
OSINT
Other
Unknown

Figure 24. Mission resource elements and intelligence information.

4. DEVELOPING A RESOURCES ARCHITECTURE

4.1. The Blue Force resources architecture

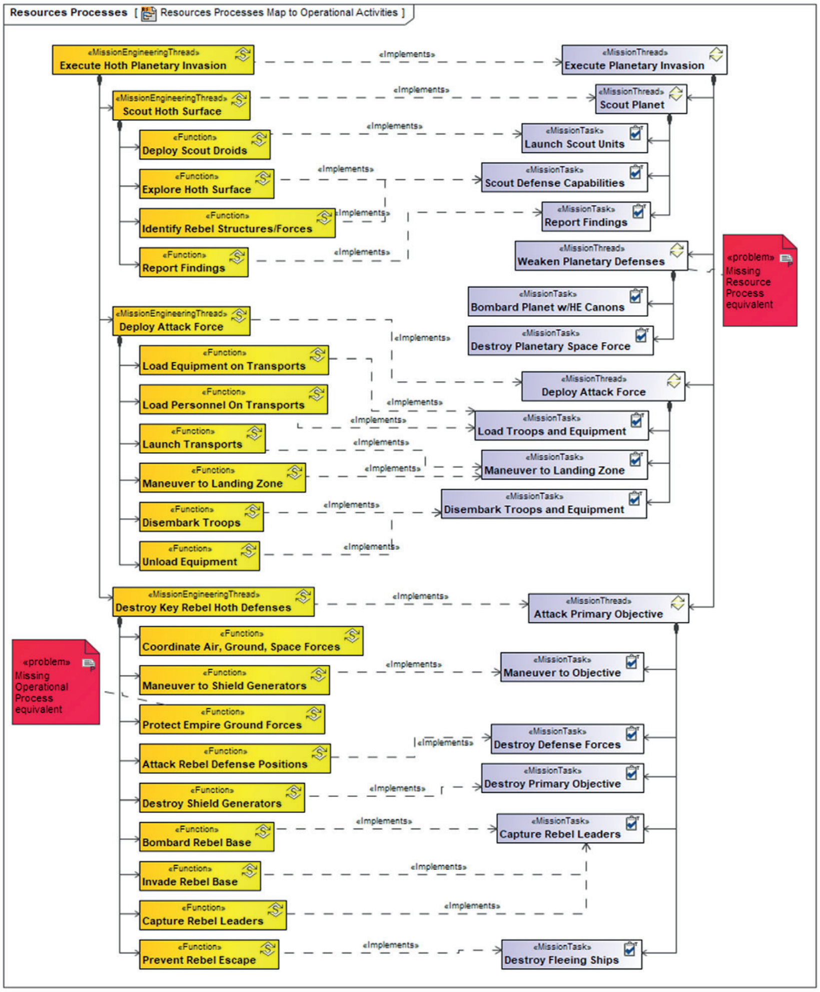
The beginnings of the Resources Architecture were seen in Figure 24, which shows a portion of the rebel forces involved in the battle. To fully understand how the mission plays out, the various resources (i.e. systems, software, personnel) that are used to support the Mission Tasks in the Mission Threads are captured in the Mission Engineering Threads (METs) that are part of the Resources Architecture. This is done for both sides of the battle – for the Blue Forces (on our side, the Empire in this case) and the Red Forces (who are the adversaries, the Rebels in the case of this example model).

4.2. Mission engineering thread functions

The various steps of the MET are defined for the Execute Hoth Planetary Invasion. The functions can either be part of the Mission Engineering Thread, or they can be Functions performed by the Resources, now that we have identified some of these Resources. Figure 25 shows the Mission Engineering Thread on the left and the Mission Thread on the right. The functions in the MET will "implement" the operational activities in the MT. This model shows two shortcomings of the architecture:

1) The MT for Weaken Planetary Defense has no corresponding resource (system) functions that implement this mission task (i.e. operational activity).

2) The MET for Destroy Key Rebel Hoth Defenses has a function Protect Empire Ground Forces that has no corresponding mission task, i.e. there is a missing operational process that needs to be performed but is not yet accounted for.



Resources Processes [Resources Processes Map to Operational Activities]
«MissionEngineeringThread» Execute Hoth Planetary Invasion
«Implements»
«MissionThread» Execute Planetary Invasion
«MissionEngineeringThread» Scout Hoth Surface
«Implements»
«MissionThread» Scout Planet
«Function» Deploy Scout Droids
«Implements»
«MissionTask» Launch Scout Units
«Function» Explore Hoth Surface
«Implements»
«MissionTask» Scout Defense Capabilities
«Function» Identify Rebel Structures/Forces
«Implements»
«MissionTask» Report Findings
«Function» Report Findings
«Implements»
«MissionThread» Weaken Planetary Defenses
«problem» Missing Resource Process equivalent
«MissionTask» Bombard Planet w/HE Canons
«MissionEngineeringThread» Deploy Attack Force
«Implements»
«MissionTask» Destroy Planetary Space Force
«Function» Load Equipment on Transports
«MissionThread» Deploy Attack Force
«Function» Load Personnel On Transports
«Implements»
«Implements»
«MissionTask» Load Troops and Equipment
«Function» Launch Transports
«Implements»
«Function» Maneuver to Landing Zone
«Implements»
«MissionTask» Maneuver to Landing Zone
«Function» Disembark Troops
«Implements»
«MissionTask» Disembark Troops and Equipment
«Function» Unload Equipment
«Implements»
«MissionEngineeringThread» Destroy Key Rebel Hoth Defenses
«Implements»
«MissionThread» Attack Primary Objective
«Function» Coordinate Air, Ground, Space Forces
«problem» Missing Operational Process equivalent
«Function» Maneuver to Shield Generators
«Implements»
«MissionTask» Maneuver to Objective
«Function» Protect Empire Ground Forces
«Function» Attack Rebel Defense Positions
«Implements»
«MissionTask» Destroy Defense Forces
«Function» Destroy Shield Generators
«Implements»
«MissionTask» Destroy Primary Objective
«Function» Bombard Rebel Base
«Implements»
«MissionTask» Capture Rebel Leaders
«Function» Invade Rebel Base
«Implements»
«Function» Capture Rebel Leaders
«Implements»
«Function» Prevent Rebel Escape
«Implements»
«MissionTask» Destroy Fleeing Ships

Figure 25. Mission resource elements and intelligence information.

4.3. Deploy scout droids process flow

Figure 26 illustrates a relatively complicated MET for the various resource elements that Deploy Scout Droids. In this process flow, the human and technical elements are deployed upon receiving the Scout OPORD. Figure 27 shows a related process flow that outlines the activities to prepare for scouting and sending back status reports.

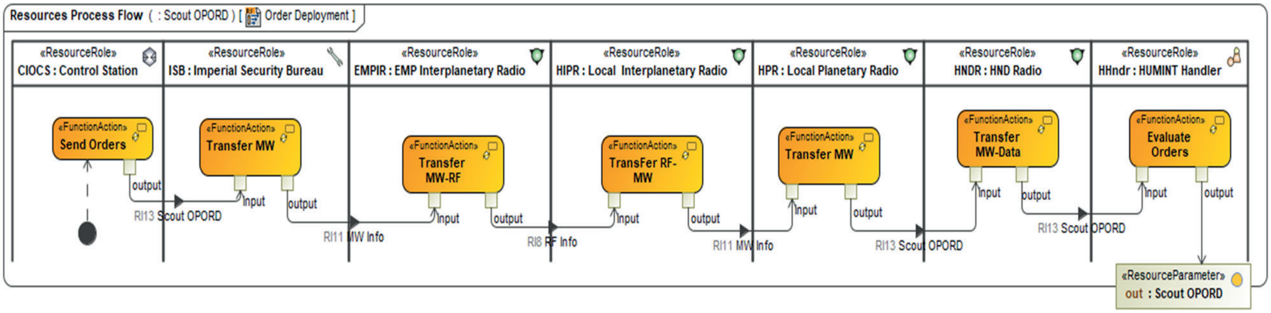

Resources Process Flow (: Scout OPORD) [Order Deployment]
«ResourceRole» CIOCS : Control Station
«ResourceRole» ISB : Imperial Security Bureau
«ResourceRole» EMPIR : EMP Interplanetary Radio
«ResourceRole» HIPR : Local Interplanetary Radio
«ResourceRole» HPR : Local Planetary Radio
«ResourceRole» HNDR : HND Radio
«ResourceRole» HHndr : HUMINT Handler
«FunctionAction» Send Orders
«FunctionAction» Transfer MW
«FunctionAction» Transfer MW-RF
«FunctionAction» TransFer RF-MW
«FunctionAction» Transfer MW
«FunctionAction» Transfer MW-Data
«FunctionAction» Evaluate Orders
output
input
RI13 Scout OPORD
RI11 MW Info
RI8 RF Info
RI11 MW Info
RI13 Scout OPORD
RI13 Scout OPORD
«ResourceParameter» out : Scout OPORD

Figure 26. Deploy scout droids process flow.

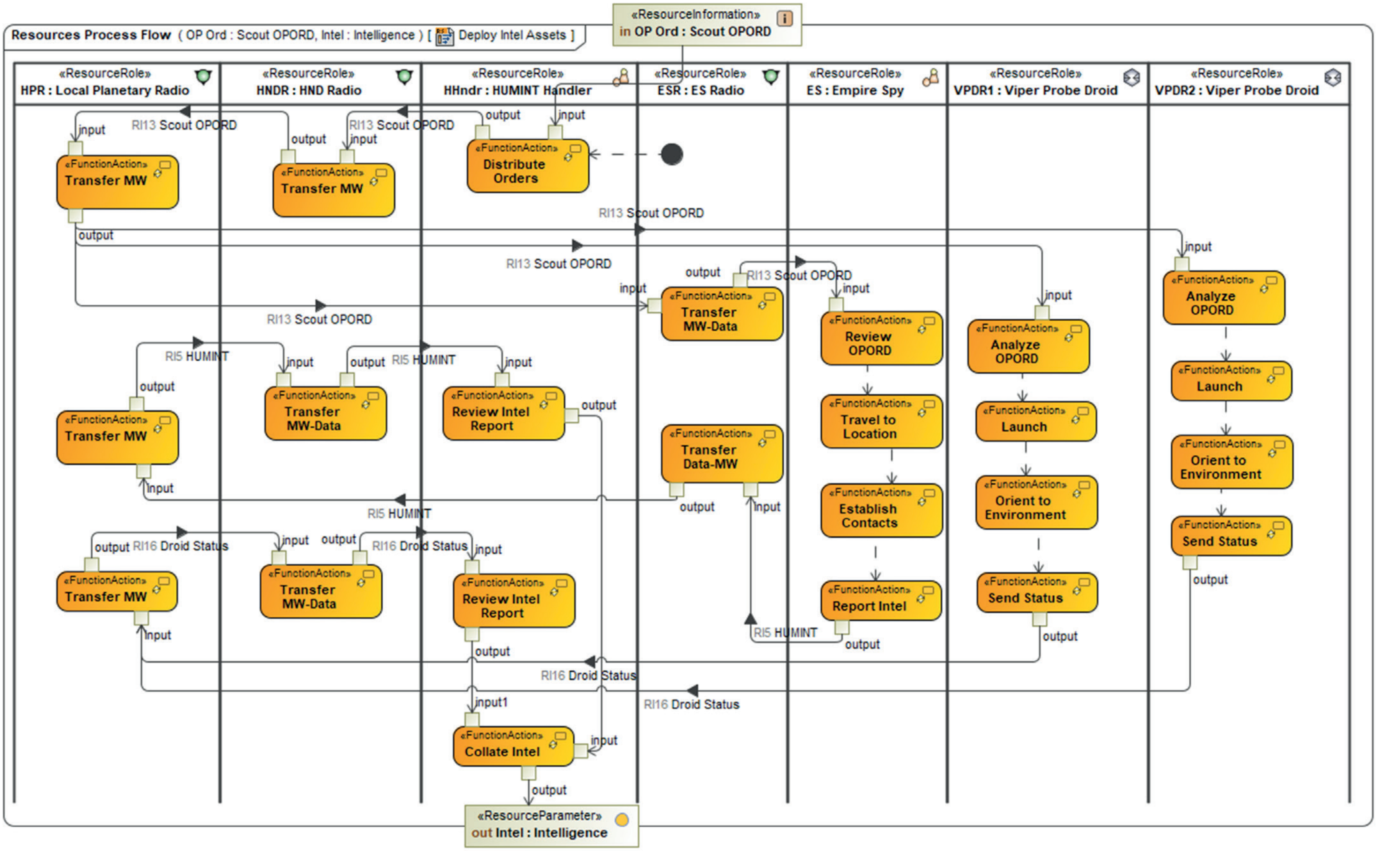

«ResourceInformation» in OP Ord : Scout OPORD
Resources Process Flow (OP Ord : Scout OPORD, Intel : Intelligence) [Deploy Intel Assets]
«ResourceRole» HPR : Local Planetary Radio
«ResourceRole» HNDR : HND Radio
«ResourceRole» HHndr : HUMINT Handler
«ResourceRole» ESR : ES Radio
«ResourceRole» ES : Empire Spy
«ResourceRole» VPDR1 : Viper Probe Droid
«ResourceRole» VPDR2 : Viper Probe Droid
«FunctionAction» Transfer MW
«FunctionAction» Transfer MW
«FunctionAction» Distribute Orders
«FunctionAction» Transfer MW-Data
«FunctionAction» Review OPORD
«FunctionAction» Analyze OPORD
«FunctionAction» Analyze OPORD
«FunctionAction» Transfer MW
«FunctionAction» Transfer MW-Data
«FunctionAction» Review Intel Report
«FunctionAction» Transfer Data-MW
«FunctionAction» Travel to Location
«FunctionAction» Launch
«FunctionAction» Launch
«FunctionAction» Establish Contacts
«FunctionAction» Orient to Environment
«FunctionAction» Orient to Environment
«FunctionAction» Transfer MW
«FunctionAction» Transfer MW-Data
«FunctionAction» Review Intel Report
«FunctionAction» Report Intel
«FunctionAction» Send Status
«FunctionAction» Send Status
«FunctionAction» Collate Intel
RI13 Scout OPORD
RI5 HUMINT
RI16 Droid Status
input
input1
output
«ResourceParameter» out Intel : Intelligence

Figure 27. Scouting and status reporting process flow.

5. OTHER CONSIDERATIONS

5.1. System of Systems issues

The systems that participate in the METs that implement the MTs in the mission architecture are usually developed by different organizations and are not always developed with these particular MTs in mind. These systems are also usually operated by different organizations involved in conducting the mission. So, from a SoS perspective, this illustrates the managerial independence and operational independence of the systems that make up the SoS for each of the METs.

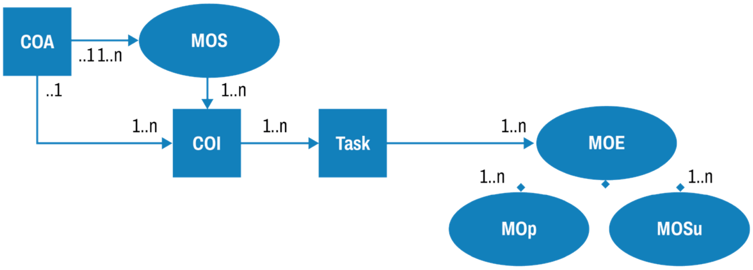


Figure 28. Relationship between critical operational issues and measures of effectiveness.

5.2. Key measurements and traceability

A well-defined model will contain quantifiable measures for success. At the very least, it will describe how to measure a successful outcome for the various mission phases and mission essential tasks. These will be modeled at the levels of the mission, mission phase, MET, and mission essential tasks. For a well-formed model, these measurements need to coincide, link, and trace to one another from system level Measures of Performance (MOPs) focused on performance of the individual constituent systems, to MOEs defining mission success at each mission essential task, to MOS defining the desired end state (see Figure 28).

These key measures must flow down from MOS and lead into the MOEs from which MOPs and Measures of Suitability (MOSu) can be derived. Though, to trace these measures through a SoS, they need to be linked to statements of importance. In this case, since we are in the military domain, we can refer to these statements as critical operational issues (COIs). For each of these COIs there may be one to many MOS, MOEs, and further one to many MOPs and MOSu as shown in Figure 28. This diagram is purely notional, as these metrics are largely stochastic, only estimable through advanced simulation, and often represent emergent properties of a complex SoS, making a deterministic 'roll-up' impractical or impossible. An example for the Hoth mission is given in Figure 29.

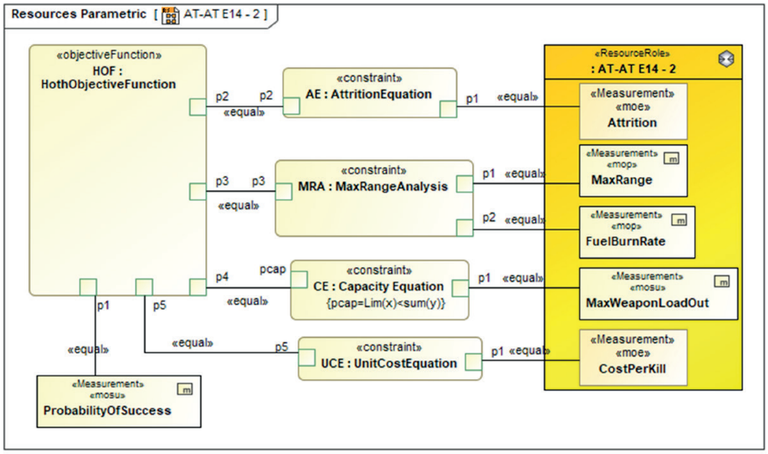


Figure 29. Typical MOEs through a mission engineering SoS lens.

5.3. All Models are Wrong, Some Models are Useful...

Even a relatively simple Resource Architecture model requires significant time and effort to develop, if everything in the architecture is modeled. As with any model, understanding what questions the model is intended to answer, what information is available to model, and what resources you have available to do the modeling (people, time, money, and tools) will help frame what needs to be modeled. Of course, once the modeling scope has been decided, any modeling scope changes must be well managed; otherwise, unintended risk to developing a useful model will be introduced.

It is likely that the entire scope of the modeling effort will not be known, as required information may not be available at the start of the modeling effort, or significant, unplanned architecture changes occur. Identifying modeling risks from the start is key to managing the modeling effort and maintaining its usefulness. It is highly recommended that prior to starting a model effort some time is spent conducting a Problem Framing exercise.

Organizations will need to determine what model libraries they want to develop, share, and maintain. Although there are many ways to separate ME models into reusable and case-specific information, UAF already segments model information such that one could simply create separate models based on the top-level packages: Strategy, Operational, Services, Personnel, Resources, Security, Projects, Standards, and Actual Resources. Of course, a model library approach will need to be made specific to how an organization wants to do modeling. A model federation plan, even just a simple one, should be devised prior to the start of modeling to help partition the large model into smaller modeling projects to facilitate model management and governance. This also helps improve time to query the model, reduce model access conflicts among team members, allow for greater control over model changes and configuration control.

6. CONCLUSIONS

Mission architecture modeling is a foundational enabler for effective ME. This chapter has demonstrated how a well-structured mission architecture, rooted in the Unified Architecture Framework (UAF), supports the rigorous analysis, traceability, and coordination required to execute complex missions involving multiple stakeholders, scenarios, and SoS. By systematically integrating strategic intent, operational activities, and resource capabilities through constructs such as MTs and METs, UAF provides a cohesive modeling language for describing and analyzing mission architectures.

Using the Battle of Hoth as an exemplar, the chapter illustrated how ME modeling not only clarifies the relationships between goals, objectives, and operational elements but also reveals risks, conflicts, and dependencies inherent in the mission. It highlighted the importance of contextual constructs such as scenarios and vignettes, and how these can be extended within UAF to provide meaningful connections between conditions, behaviors, and mission execution.

Moreover, the use of measures (MOS, MOEs, MOPs) and the emphasis on traceability from high-level drivers to tactical resource activities ensure that the mission model can serve as a foundation for analysis, assessment, and iterative improvement. The incorporation of provenance and confidence in intelligence data, compliance to doctrine, and consideration of force designations further enhances model fidelity and realism.

Ultimately, mission architecture modeling is not about producing a perfect representation, but rather about building a useful one; one that can inform decisions, identify vulnerabilities, and guide effective action. The principles and constructs presented in this chapter offer a scalable and repeatable approach for applying mission architecture modeling across defense, aerospace, and enterprise contexts.

REFERENCES

1. OMG 2022, Unified Architecture Framework, Version 1.2, Object Management Group, https://www.omg.org/spec/UAF/About-UAF/.

2. OMG 2022, Unified Architecture Framework Modeling Language, Version 1.2, Object Management Group.

3. OMG 2022, Enterprise Architecture Guide for the Unified Architecture Framework (Informative), Version 1.2, Object Management Group.

4. Gagliardi M., Hause M., "Implementing Mission Engineering with UAF" In Proceedings of the Ground Vehicle Systems Engineering and Technology Symposium (GVSETS), NDIA, Novi, MI, Aug. 16-18, 2022.

5. Gagliardi M., Hause M., Martin J., Phillips, M, 2024, "Darth Vader's Secret Weapon: Implementing Mission Engineering with UAF" presented at the INCOSE International Symposium, Dublin, Ireland.

6. Martin, J., & Alvarez, K., 2023, "Using the Unified Architecture Framework in support of mission engineering activities" presented at the INCOSE International Symposium.

7. Fandom, 2023, Battle of Hoth, Reference information Available online from https://starwars.fandom.com/wiki/Star_Wars:_Episode_V_The_Empire_Strikes_Back#The_Battle_of_Hoth

8. DoD, 2023, DoD OSD Mission Engineering Guide (MEG), Available online at https://ac.cto.mil/wp-content/uploads/2023/11/MEG_2_Oct2023.pdf Accessed November 2023.

BIOGRAPHIES

MATTHEW GAGLIARDI

Matthew Gagliardi is a Principal at SSI and has over twenty years of experience in Systems Engineering in the Automotive and Defense sectors. His expertise includes vehicle development, system integration, systems engineering processes and tools, and new technology development. Matt currently provides MBSE support and guidance to multiple PEO Ground Combat Systems programs, as well as to DEVCOM's Ground Vehicle Systems Center. Matt received a Bachelors in Mechanical Engineering and a MS in Mechanical Engineering at Purdue University. Matt has achieved SPRDE Level 3 from Defense Acquisition University and is an OMG Certified Systems Modeling Professional Model Builder and has been an INCOSE CSEP. His role at SSI includes Consulting, mentoring, conference presentations, and developing and presenting training courses.

MATTHEW HAUSE

Matthew Hause is Principal Engineer at SSI, a chair of the UAF group, a member of the OMG SysML specification team, a co-chair of the OMG Reusable Asset Specification (RAS) team, and a thought leader in the use of MBSE. He has been developing multi-national complex systems for over 45 years as a systems and software engineer. He started out working in the power systems industry then transitioned to command and control systems, process control, communications, SCADA, distributed control, military systems, and many other areas of technical and real-time systems. He has authored over 100 technical papers on a diverse range of subjects. His role at SSI includes Consulting, mentoring, standards development, presentations at conferences, and developing and presenting training courses. He is also a proud recipient of the INCOSE MBSE Propeller Hat Award.

DR. JAMES N MARTIN

Dr. James N. Martin is an Enterprise Architect and a Distinguished Systems Engineer at The Aerospace Corporation developing solutions for information systems and for space domain enterprises. He is a member of the UAF Revision Task Force with OMG and was lead editor for the ISO 42020 standard on Architecture Processes. Dr. Martin was a key author on the BKCASE project in development of Enterprise Systems Engineering articles for the SE Body of Knowledge (SEBOK). He led the working group responsible for developing ANSI/EIA 632, a US national standard that defined the processes for engineering a system. He previously worked for Raytheon Systems Company and AT&T Bell Labs on airborne and underwater systems and on mobile telecommunication systems. His book, Systems Engineering Guidebook, was published by CRC Press in 1996. Dr. Martin is an INCOSE Fellow and was leader of the Standards Technical Committee. He was the founder and was until recently leader of the Systems Science Working Group. He received from INCOSE the Founders Award for his long and distinguished achievements in the field.

VÍCTOR RAMOS DEL POZO

Víctor Ramos del Pozo is a systems engineer at Isdefe, currently delivering technical assistance for the Spanish National Programme Office for the Next Generation Weapon System (NGWS) within a Future Combat Air System (FCAS), especially in the fields of the Combat Cloud, Collaborative Sensors, Simulation and Systems of Systems. He previously delivered technical assistance for the Spanish National Armaments Directorate of the State Secretary for Defence in the fields of defence industrial base analysis, defence acquisition planning and major defence acquisition programmes management.

Before joining Isdefe, he worked as a systems engineer at EADS-CASA (current Airbus Defence and Space) for the Multi-Role Tanker Transport and Future Strategic Tanker Aircrafts programmes, and at Indra for the Identification Friend or Foe Department.

He is a fellow of the International Council on Systems Engineering (INCOSE) and delivers internal training at Isdefe in the fields of systems engineering and programme management.

In 2015 he was awarded with the Spanish Air Force Medal (Cruz del Mérito Aeronáutico con distintivo blanco) for his work supporting the Spanish National Armaments Directorate.

1000
RX
1250_5825_2563_2541
100
RX
050
RX
001
RX
001
TX
002
TX
003
TX
1250_5825_2563_2541
250
200
001
060 %
150
INPUT
A
B
C
AC_251454
36259
55874
12345
44864555
14526522
64555
12345
1250_5825_2563_2541
20

CHAPTER 5

Design and implementation of SoS governance

Dr. Polinpapilinho F. Katina, *University of South Carolina Upstate (pkatina@uscupstate.edu)*
Dr. Charles B. Keating, *Old Dominion University (ckeating@odu.edu)*
César Heras Menor de Gaspar, *Isdefe (chmenor@isdefe.es)*
Víctor Ramos del Pozo, *Isdefe (vramos@isdefe.es)*

Abstract

Given the independence and autonomy of the constituent systems of a System of Systems (SoS), governance becomes a critical aspect of their operation. At a basic level, governance provides for the direction, oversight, and accountability of a SoS in fulfilling its intended purpose/mission. This chapter provides an introductory treatment of the design and implementation of SoS governance. Three primary topics are articulated. First, a systems-based perspective of SoS governance is provided. Second, a next-generation systems theory-based framework for SoS governance, Complex System Governance, is established. Third, the application of the framework for SoS governance is explored. The chapter concludes with challenges and application guidance for practitioners responsible for SoS design and governance implementation.

Keywords

SoS Governance, Framework, Systems Theory, Complex System Governance

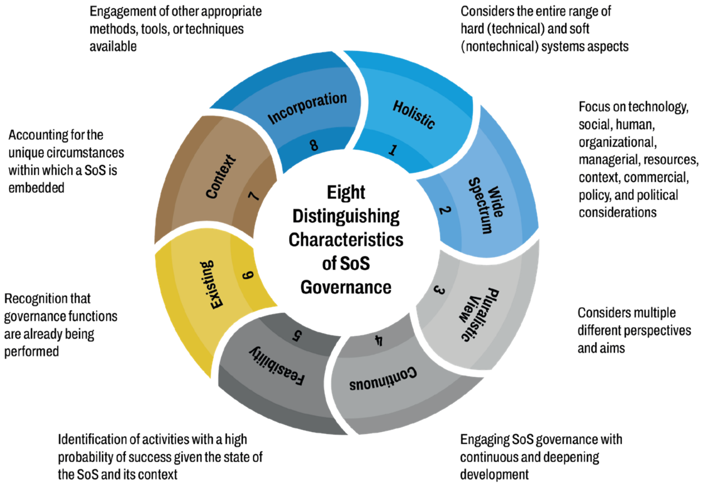
1. INTRODUCTION

The focus on 'governance' for SoS is targeted to the active steering of a SoS through the artful and integrated design, execution, and evolution of the SoS [1]. Thus, the primary motivation for this chapter is to present the role and contributions that governance can make to the design and implementation of SoS.

In this chapter, we address three primary topics. First, we articulate a systems-based perspective of SoS governance. This perspective examines the nature and role of governance with respect to SoS governance design and implementation. Second, Complex System Governance (CSG) is introduced as a systems theory-based framework for SoS governance. CSG is established as an evolution of System of Systems Engineering, focused on providing an approach to support more effective direction, oversight, and accountability for increasingly complex SoS. Third, the application of the framework for SoS governance is explored. The chapter concludes with challenges and application guidance for practitioners responsible for SoS design and governance implementation.

In evolving SoS design and implementation, eight distinguishing characteristics for the governance of SoS [2] are offered (Figure 1):

1. **Holism:** Governance operates at all levels of an SoS, ranging from individual practitioner to enterprise. Consideration is given to avoiding both hard (technical) and soft (nontechnical) failure modes.
2. **Wide Spectrum:** The governance focus for SoS looks beyond the narrow confines of technology-based solutions. Instead, the scope of governance entails technology, social, human, organizational, managerial, resources, context, commercial, policy, and political considerations.
3. **Pluralistic View:** Governance does not assume a unitary (singular agreement) view of SoS design and implementation. Instead, a pluralist (multiple perspectives) view is taken to understand that SoS governance must contend with numerous different and potentially conflicting motivations, perspectives, and aims.
4. **Continuous:** Governance development is continuous, realizing that SoS governance is never complete and must be continually pursued.
5. **Feasibility:** Governance development must consider that there are design and implementation issues that lie beyond the capacity of a SoS to address. Every SoS has limitations as to what can be pursued with a high possibility of success.
6. **Existing:** Irrespective of their acknowledgment, the functions of SoS governance are being performed by any viable (continuing to exist) SoS. However, the state and performance of the SoS governance may fall short of that desire and potentially be moving toward failure.
7. **Context:** Context includes those circumstances, factors, and conditions that influence, and are influenced by the SoS. The context within which SoS governance is performed is critical to enabling or constraining SoS governance.
8. **Incorporation:** Design and implementation of SoS governance must incorporate the variety of methods, tools, or techniques available and accessible to support governance development. Every SoS is unique and will have different methods, tools, and techniques available.



Engagement of other appropriate methods, tools, or techniques available
Considers the entire range of hard (technical) and soft (nontechnical) systems aspects
Focus on technology, social, human, organizational, managerial, resources, context, commercial, policy, and political considerations
Accounting for the unique circumstances within which a SoS is embedded
Recognition that governance functions are already being performed
Considers multiple different perspectives and aims
Identification of activities with a high probability of success given the state of the SoS and its context
Engaging SoS governance with continuous and deepening development
Incorporation
8
Holistic
1
Wide Spectrum
2
Pluralistic View
3
Continuous
4
Feasibility
5
Existing
6
Context
7
Eight Distinguishing Characteristics of SoS Governance

Figure 1. Eight distinguishing characteristics of SoS governance.

2. GOVERNANCE IN THE CONTEXT OF SYSTEM OF SYSTEMS

For SoS, governance plays a critical role in ensuring consistency in the direction, oversight, and accountability as the SoS performs its mission. In this section, we distinguish between management and governance and explore the role of governance in SoS design and implementation.

2.1. Nature and role of governance

The concept of governance is somewhat novel with respect to SoS. At a high level, the focus on 'governance' is taken as the active steering of a system through the artful and integrated design, execution, support, and evolution of the SoS [2].

Governance has similarities but is also different than management. Table 1 identifies the critical distinctions between management and governance along with implications for SoS design and implementation of governance [2, 3].

Characteristic	Management	Governance	Implications for SoS Design and Implementation
Emphasis	Outputs (tangible, objective, short-term)	Outcomes (less tangible, subjective, long term)	Determination of governance 'goodness' is not simple or straightforward.
Central questions of concern	What? And How?	Why?	Governance exists at a higher logical level of performance – emphasizing system purpose.
Focus	Near-term demonstrable results	Long term future focused trajectory	The focus of governance is expansive, entertaining long-view questions of strategic rather than operational significance.
Determinants of success	Easily defined, measured, and tracked	Difficult to define and measure	While governance measures might be developed, they necessarily lack precision.
Time horizon	Short term	Long term	The nature of governance invokes a much longer time horizon.
Action-response proximity	Limited duration between action and system response	Long separation between action and realization of system response	The evolutionary nature of SoS lengthens the time and proximity between actions and realization of results.
Uncertainty	Local uncertainty concerns	Global uncertainty concerns	Governance has a more global level of uncertainty and its resolution.
Stability and emergence	Local proximity stability, local level emergence	Global proximity stability, global level emergence	The global focus of governance is concerned with emergence and stability at a higher level.

Table 1. Differences between management and governance.

Etymologically, 'governance' (govern) finds its roots in the Latin '*gubernare*' meaning to steer, as well as the Greek '*kybernētēs*', meaning to pilot, governor (from Hibernian to steer, govern). Thus, at the most fundamental level, governance is about steering.

There are multiple perspectives on governance. However, for SoS, three perspectives are essential concerning the role of governance [3]:

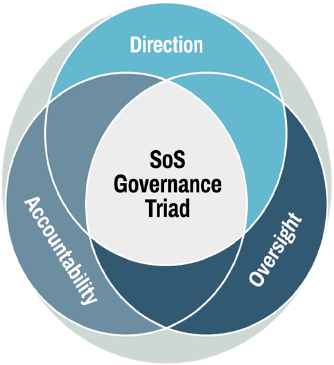
1. **Process-centric:** Governance is focused on collective decision-making processes. These processes are steeped in formal, consensus-seeking, and deliberative decision-making. The central objective of this governance perspective is to provide effective processes. These processes enable the act(s) of governance to be performed.
2. **Structure-centric:** Governance emphasizes the formulation and execution of structures. These structures preserve order/continuity while steering the system in the desired direction. The primary objective is to establish sufficient structure such that the trajectory of a system toward desired ends is maintained.
3. **Policy-centric:** Governance emphasizes the formulation of policies. These policies are targeted to inculcate the rules, norms, principles, and behaviors that support regularity in performance. The primary objective is to invoke policies that support direction/control essential to achieving/maintaining system performance.

2.2. The unique role of governance for SoS design and implementation

Based on this discussion of governance, we can draw several important themes for SoS design and implementation. For SoS, governance must provide continuous achievement of a triad (see Figure 2):

1) *Direction:* sustainment of a coherent identity (grounding essence of an SoS) that is capable of supporting consistency in the decision, action, interpretation in support of long-range strategic trajectory of the SoS;
2) *Oversight:* providing necessary controls, regulation, and performance monitoring to integrate the SoS in performance of the mission/purpose; and
3) *Accountability:* ensuring that responsibilities are established and resources are efficiently and effectively utilized to support achievement of SoS aims.

With these three pillars of governance, the achievement of the *Process-Structure-Policy* imperatives for SoS design and implementation of governance can be developed.



Provision of steering for the SoS trajectory - integration of constituents for consistency in the achievement of purpose/mission through process, structure, and policy.

Ensuring efficiency in the execution of process, structure, and policy - detection and resolution of anomalies in governance performance.

Monitoring the performance of the SoS- assuring that direction is maintained and process, structure, and policy are effectively implemented.

Figure 2. The SoS governance triad.

3. SOS GOVERNANCE

Present-day SoS are increasingly interconnected and complex yet enable possibilities far beyond any previous capabilities. These enhanced capabilities exist beyond SoS member systems and could not have been imagined a decade ago. It almost goes without saying that we are experiencing difficulties in governing SoS as we seem incapable of matching the acceleration of information, interconnectedness, and technology driving our current state of affairs. For all the 'goodness' and capabilities that SoS have brought, they have also generated problems that appear to be intractable given our current methods and frameworks to address them [1-3]. The time is appropriate for designing and implementing new frameworks to enhance our ability to more effectively govern present and future evolutions of SoS.

Complex System Governance (CSG) is an emerging evolution of SoS seeking to enhance our capabilities for the design, execution, and evolution of SoS. The emerging CSG field offers a new and novel path forward in governing increasingly complex SoS and their problems. Since its inception in 2014, CSG has matured significantly, evolving from its early stages into a knowledge-rich field, contributing both theoretical insights and practical applications for operational SoS [4].

CSG evolved from the work in System of Systems Engineering (SoSE) as articulated by [5, 6]. CSG's development was partly due to the limitations of SoSE to consider both the 'hard' (technical) and 'soft' (nontechnical) aspects of complex systems, moving beyond the strict dominance of 'technology first, technology only' based approaches and solutions. Thus, by incorporating the essence of Systems theory, CSG aimed to address the entire spectrum of SoS, spanning the entire range of socio-technical-economic-political dimensions. Additionally, incorporating governance moved CSG to a higher level of emphasis, focused on direction, oversight, and accountability across the entire range of socio-technical-economic-political drivers of SoS. The CSG field is very young, and while much remains to be accomplished as CSG continues to develop, the time is appropriate to acknowledge and amplify CSG's contributions to SoS governance.

In this section, we provide a framework for designing and implementing SoS governance (CSG). The section begins by explaining the two informing fields —Management Cybernetics and Systems Theory— which, alongside (System) Governance, provide the conceptual foundations for the establishment of CSG. Next, the CSG Paradigm is articulated to establish the fundamental aspects of CSG for SoS governance. Following the paradigm introduction, the detailed framework for CSG is provided. This framework, built upon the underlying informing fields and paradigm, provides the essence of CSG through a set of functions and associated communication channels. Together, these provide for the design and implementation of governance for a SoS.

3.1. SoS governance at the intersection of three fields

Three informing fields serve as the conceptual basis for CSG [3, 30]. In broad terms, these fields include:

1) **General Systems Theory,** which provides the set of propositions (laws, principles, concepts) that serve to define the behavior, structure, and performance of all complex systems;

2) **Management Cybernetics,** which is described as the science of effective system structural organization targeted to assure system viability (continued existence); and

3) **System Governance,** which is focused on establishing direction, oversight, and accountability for complex systems. System governance has been discussed above. We continue with a discussion of the two remaining fields, which will be briefly examined for their unique contribution to CSG as a foundation for the design and implementation of SoS governance.

3.1.1. Systems theory contributions

The contribution of Systems Theory to SoS Governance is threefold. First, Systems Theory provides a strong and rigorous theoretical grounding. Second, Systems Theory has been articulated as a set of axioms (taken-for-granted assumptions) and associated propositions (principles, laws, and concepts) that seek to describe, explain, and predict the behavior, structure, or performance of systems, either natural or manmade [7-9]. While a detailed development of Systems Theory is beyond the scope of this chapter, it suffices to say that the axioms and propositions provide the intellectual basis for SoS governance. Third, Systems Theory provides a language to understand systemic deficiencies that impede the performance of governance functions for SoS. These deficiencies, known as pathologies, are observed as violations of underlying Systems Theory propositions.

In effect, Systems Theory provides a theoretical grounding for the design and implementation of SoS governance, such that integration and coordination necessary to ensure SoS viability can be maintained.

3.1.2. Management cybernetics contributions

Management Cybernetics is broadly defined as 'the science of effective system structural organization' [10-12]. Critical to CSG is the concept of the 'metasystem' as a set of interrelated functions that must be performed by any viable (continuing to exist) system. The metasystem provides integration (allowing a system to act as a unity) and coordination (providing for smooth interaction among system constituents). Thus, a SoS is structured in a way that permits it to meet performance levels necessary to continue to be viable (exist).

Management Cybernetics brings three important contributions to the design and implementation of governance for SoS. First, grounding in Management Cybernetics offers a strong theoretical/conceptual foundation. Management Cybernetics, at a most basic level, is concerned with communication and control. This aligns with governance to provide direction and monitoring as a system continues on a desirable trajectory. Concerning control, the cybernetic viewpoint suggests that control-based constraints placed on a system provide regulatory capacity essential to assure system performance and continued viability. Second, Management Cybernetics forms a basis for governance structure, which includes functions and communication channels consistent with the achievement of governance for a system [10-13]. Management Cybernetics is a launching point for the CSG Reference Model [3]. Third, Management Cybernetics has been successfully applied for over five decades. Despite technological shifts and the rapid pace of change in SoS, Management Cybernetics has remained relevant and maintained a strong and sustained presence.

3.2. SoS governance – A next generation paradigm

CSG is the "Design, execution, and evolution of the metasystem functions necessary to provide control, communication, coordination, and integration of a complex system" [4, p. 264]. Within this definition, we find the underlying paradigm that brings the definition to life [2, 14]. The paradigm that instantiates this definition is captured in Figure 3.

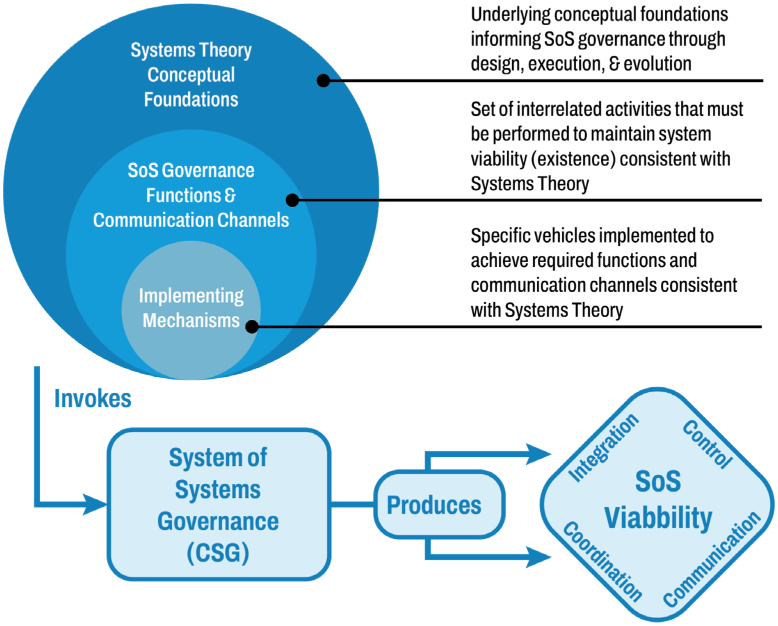

Systems Theory Conceptual Foundations
SoS Governance Functions & Communication Channels
Implementing Mechanisms
Underlying conceptual foundations informing SoS governance through design, execution, & evolution
Set of interrelated activities that must be performed to maintain system viability (existence) consistent with Systems Theory
Specific vehicles implemented to achieve required functions and communication channels consistent with Systems Theory
Invokes
System of Systems Governance (CSG)
Produces
Integration
Control
SoS Viabbility
Coordination
Communication

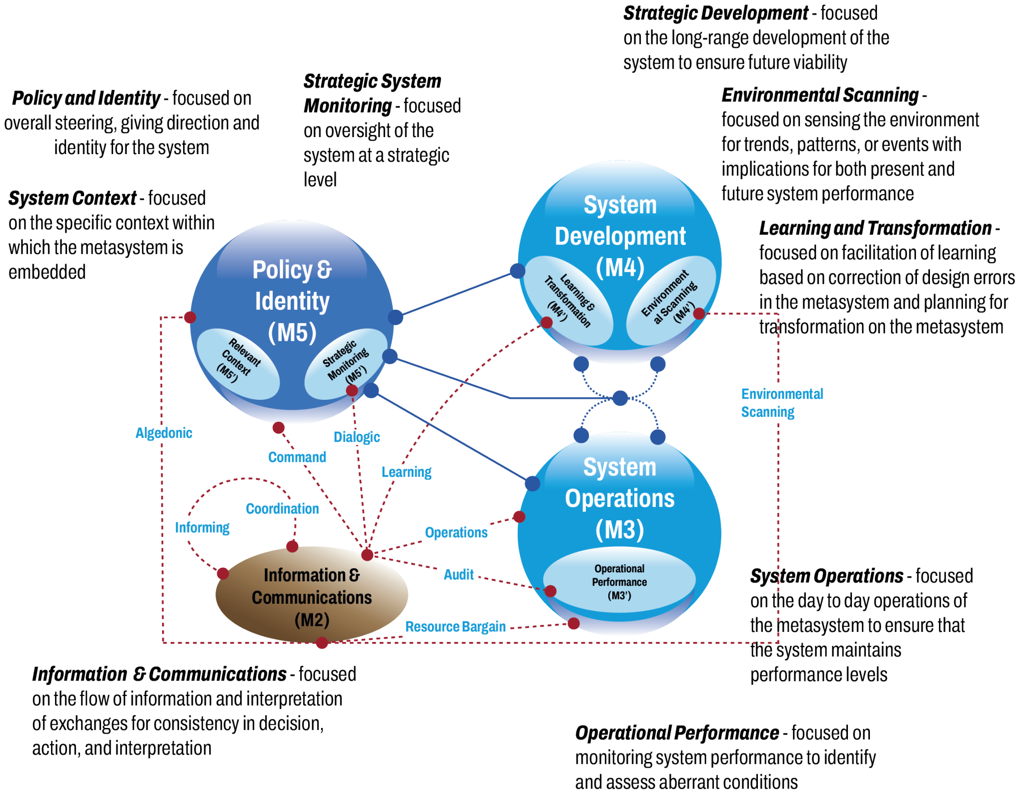
Figure 3. The SoS governance paradigm relationships.

First, *design* accentuates the necessity to purposely pursue the creation of the governance structure. While the design for SoS governance represents the normative case, execution tempers the normative design based on deployment in the operational setting. Where design meets execution, the result is inevitably a design that requires *evolution (development)* to make modifications necessary to adjust to unknowns, emergence, and design inadequacies for a given context of application.

Second, the four aspects of CSG include control (the regulatory constraints that ensure SoS performance and trajectory), *communication* (the flow, processing, and interpretation of information through channels), *coordination* (focused on interaction among constituent entities comprising the system, and with those external to the system, to prevent unnecessary fluctuations), and *integration* (maintenance of system unity through common goals, accountability, and balance between individual constituent autonomy and system level interests).

Third, viability (continued existence) is assured by the performance of *functions* (system imperatives that must be performed to maintain viability) and associated *communication channels* (the conduits that provide for the flow of information and interpretation within and external to the system). The functions and communication channels together comprise the *'metasystem'* which institutes SoS governance (Figure 4).

Fourth, functions and communication channels are performed by *mechanisms* (vehicles that serve to implement) that are unique to each system governed.



Policy and Identity - focused on overall steering, giving direction and identity for the system

Strategic System Monitoring - focused on oversight of the system at a strategic level

Strategic Development - focused on the long-range development of the system to ensure future viability

Environmental Scanning - focused on sensing the environment for trends, patterns, or events with implications for both present and future system performance

System Context - focused on the specific context within which the metasystem is embedded

Learning and Transformation - focused on facilitation of learning based on correction of design errors in the metasystem and planning for transformation on the metasystem

System Operations - focused on the day to day operations of the metasystem to ensure that the system maintains performance levels

Information & Communications - focused on the flow of information and interpretation of exchanges for consistency in decision, action, and interpretation

Operational Performance - focused on monitoring system performance to identify and assess aberrant conditions

Figure 4. Metasystem functions and associated communication channels.

Central to SoS governance is the metasystem. The metasystem is the composite of functions and communication channels that are above and beyond individual systems that enable SoS governance. Nine interrelated functions and ten communication channels capture the essence of governance for a SoS [3, 4, 15]. These functions are an extension of Management Cybernetics [10-12]. The metasystem functions and corresponding communication channels are depicted in Figure 4. As the figure shows, there are four primary functions and five related subordinate functions, along with 10 communication channels [2-4, 16].

CSG is the set of 9 interrelated functions that act to provide governance for a complex system. These functions include:

- **Metasystem Five (M5) – Policy and Identity:** provides overall steering (e.g., vision, purpose, mission) for the SoS, giving direction to ensure that the trajectory of the SoS is retained, provides for maintenance of identity (the essence of uniqueness for the SoS) responsible to engender consistency in decision, action, and interpretation, represents the SoS to the 'outside', maintains boundary conditions, and balances focus between short and long term SoS interests.
- **Metasystem Five Star (M5*) – System Context:** responsible for elaborating and managing the specific context (factors that enable and constrain the performance of the SoS, e.g., support infrastructure, culture, stakeholders) within which the metasystem is embedded. Monitors and facilitates communication of contextual factors, within and external to the SoS.
- **Metasystem Five Prime (M5') – Strategic System Monitoring:** provides oversight of the system performance at a strategic level and determines the degree to which the SoS is effective in pursuit of long-range directions and maintenance of future trajectory.
- **Metasystem Four (M4) – System Development:** emphasizes understanding and implications for pursuing and achieving the long-range development of the SoS to ensure future viability. Processes environmental scanning to determine impacts on present operations and future development.
- **Metasystem Four Star (M4*) – Learning and Transformation:** concentrated on facilitating learning based on correcting design errors in the metasystem and planning for the responsive transformation of the SoS.

- **Metasystem Four Prime (M4') – Environmental Scanning:** designs, deploys and monitors sensing of the environment for trends, patterns, conditions, circumstances, or emergent events with implications for both present and future system viability. Maintains an active model of the SoS environment.

- **Metasystem Three (M3) – System Operations:** focuses on the day-to-day operations of the metasystem to ensure that the SoS maintains desired performance levels consistent with resource distributions to produce value by the SoS.

- **Metasystem Three Star (M3*) – Operational Performance:** concerned with developing and monitoring system operational performance measures to monitor productivity achievement and identify and assess aberrant conditions.

- **Metasystem Two (M2) – Information and Communications:** focused on the design for the flow and interpretation of information within the SoS metasystem and from the SoS metasystem to the constituent systems. Provides for consistent interpretation of exchanges through communication channels to support consistency in decision, action, and interpretation within and external to the SoS.

One way the CSG's nine interrelated functions enable SoS governance is through the *metasystem communication channels.* These channels support the flow of information for decision and action as well as produce consistency in interpretation for exchanges within the metasystem and between the metasystem and external entities. Table 1 below concisely lists the communication channels, their primary associated metasystem function, and their particular role in SoS governance.

Communications Channel and Associated Metasystem Function	Channel Role for SoS Governance
Command (Metasystem 5)	• Provides non-negotiable direction to the metasystem and governed systems. • Primarily flows from the Metasystem 5 and is disseminated throughout the system.
Resource bargain/ Accountability (Metasystem 3)	• Determines and allocates the resources (manpower, material, money, methods, time, information, support) to governed systems. • Defines performance levels (productivity), responsibilities, and accountability for governed systems. • Primarily an interface between Metasystem 3 to the governed systems.
Operations (Metasystem 3)	• Provides for the routine interface concerned with near-term operational focus. • Concentrated on providing direction for system production of value (products, services, processes, information) consumed external to the system. • Primarily an interface between Metasystem 3 and governed systems.
Coordination (Metasystem 2)	• Provides for metasystem and governed systems balance and stability. • Ensures design and achievement (through execution) of design: (1) sharing of information within the system necessary to coordinate activities, and (2) ensures decisions and actions necessary to prevent disturbances are shared within the Metasystem and governed systems. • Primarily a channel designed and executed by Metasystem 2.
Audit (Metasystem 3*)	• Provides routine and sporadic feedback concerning operational performance. • Investigation and reporting on problematic performance issues within the system. • Primarily a Metasystem 3* channel for communicating between Metasystem 3, the governed systems, and the metasystem concerning performance issues.

Communications Channel and Associated Metasystem Function	Channel Role for SoS Governance
Algedonic (Metasystem 5)	• Provides a ‘bypass’ of all channels when the integrity of the system is threatened. • Compels instant alerts to crisis or potentially catastrophic situations for the system. • Directed to Metasystem 5 from anywhere in the metasystem or governed systems.
Environmental Scanning (Metasystem 4’)	• Provides design for sensing to monitor critical aspects of the external environment. • Identifies environmental patterns, activities, or events with system implications. • Provided for access throughout the metasystem as well as governed systems by Metasystem 4’.
Dialog (Metasystem 5’)	• Provides for examination of system decisions, actions, and interpretations for consistency with system purpose and identity. • Directed to Metasystem 5’ from anywhere in the metasystem or governed systems.
Learning (Metasystem 4*)	• Provides detection and correction of error within the metasystem as well as governed systems, focused on system design issues as opposed to execution issues. • Directed to Metasystem 4* from anywhere in the metasystem or governed systems.
Informing (Metasystem 2)	• Provides for flow and access to routine information within the metasystem or between the metasystem and governed systems. • Access provided to the entire metasystem and governed systems. • Primarily designed by Metasystem 2 for utilization by all metasystem functions as well as governed systems.

Table 2. Communication channels to support SoS governance.

At first exposure to the framework for SoS Governance, it appears somewhat detailed and perhaps overwhelming. However, the functions and communication channels are already being performed to some degree in each viable (continuing to exist) SoS. Unfortunately, they are most likely performed in a piecemeal (ad hoc) fashion, without the benefits of purposeful and integrated design, oversight, and accountability. Ultimately, a SoS may be governed without explicit acknowledgment of the functions and communication channels. However, we suggest that if the performance of SoS governance is to reach higher levels, the functions and communication channels offer an explicit framework to engage in rigorous self-study and development.

4. APPLICATION FOR DESIGN AND IMPLEMENTATION OF SOS GOVERNANCE

Given the SoS Governance framework provided, our question now becomes, How can this framework be implemented to support the purposeful development of SoS governance? To answer this question, we focus on the identification of SoS governance deficiencies. Governance deficiencies are identified as ‘pathologies’, which are aberrations in normal or healthy performance of governance functions. This section examines the question of SoS governance implementation with three central topics. First, the identification of pathologies, across the metasystem governance functions is explored. These pathologies indicate areas where governance functions fall short of meeting desirable performance expectations. Second, several scenarios are explored where SoS governance might offer utility for improving the current and future SoS performance. Third, a set of practitioner guidance for the implementation of SoS governance is developed. This guidance offers ‘lessons’ from our applications of CSG to improve the state of governance for SoS.

4.1. Identification of pathologies as issues in SoS governance

At a basic level, pathology is generally associated with health, where pathology indicates a departure from what is expected as normal or healthy system operation (e.g., the human body). With respect to SoS governance, a SoS pathology is "A circumstance, condition, factor, or pattern that acts to limit system performance, or lessen system viability, such that the likelihood of a system achieving performance expectations is reduced" [18, p. 253]. In essence, a pathology is the degradation of a system function, impacting the ability of the system to produce desirable performance. Pathology is observable as symptomatic of an underlying condition. Thus, a pathology is not necessarily something obvious. Instead, it requires exploration at a deeper systemic level beyond its surface-level symptomatic manifestation. In SoS Governance, a pathology is indicative of 'violations' of Systems Theory propositions (laws, principles, and concepts).

CSG functions and communication channels that provide for SoS governance offer a set of "coordinates" to locate the existence of a pathology. This location is linked to the nine different metasystem functions essential to the continued viability of a SoS, which are articulated as a set of 53 specific pathologies in relationship to the metasystem functions are articulated. These pathologies are organized around the nine metasystem functions and serve to identify aberrations to the normal (healthy) performance of metasystem functions (Table 3) [18, 19].

Metasystem function	Corresponding set of pathologies
Metasystem five (M5): Policy and identity	M5.1. Identity of system is ambiguous and does not effectively generate consistency system decision, action, and interpretation.
	M5.2. System vision, purpose, mission, or values remain unarticulated, or articulated but not embedded in the execution of the system.
	M5.3. Balance between short term operational focus and long term strategic focus is unexplored.
	M5.4. Strategic focus lacks sufficient clarity to direct consistent system development.
	M5.5. System identity is not routinely assessed, maintained, or questioned for continuing ability to guide consistency in system decision and action.
	M5.6. External system projection is not effectively performed.
Metasystem Five Star (M5*): System context	M5*.1. Incompatible metasystem context constraining system performance.
	M5*.2. Lack of articulation and representation of metasystem context.
	M5*.3. Lack of consideration of context in metasystem decisions and actions.
Metasystem Five Prime (M5'): Strategic system monitoring	M5'.1. Lack of strategic system monitoring.
	M5'.2. Inadequate processing of strategic monitoring results.
	M5'.3. Lack of strategic system performance indicators.
Metasystem Four (M4): System development	M4.1. Lack of forums to foster system development and transformation.
	M4.2. Inadequate interpretation and processing of results of environmental scanning – non-existent, sporadic, limited.
	M4.3. Ineffective processing and dissemination of environmental scanning results.
	M4.4. Long-range strategic development is sacrificed for management of day-to-day operations – limited time devoted to strategic analysis.
	M4.5. Strategic planning/thinking focuses on operational level planning and improvement.

Metasystem function	Corresponding set of pathologies
Metasystem Four Star (M4*): Learning and transformation	M4*.1. Limited learning achieved related to environmental shifts.
	M4*.2. Integrated strategic transformation not conducted, limited, or ineffective.
	M4*.3. Lack of design for system learning – informal, non-existent, or ineffective.
	M4*.4. Absence of system representative models – present and future.
Metasystem Four Prime (M4'): Environmental scanning	M4'.1. Lack of effective scanning mechanisms.
	M4'.2. Inappropriate targeting/undirected environmental scanning.
	M4'.3. Scanning frequency not appropriate for rate of environmental shifts.
	M4'.4. System lacks enough control over variety generated by the environment.
	M4'.5. Lack of current model of system environment.
Metasystem Three (M3): System operations	M3.1. Imbalance between autonomy of productive elements and integration of whole system.
	M3.2. Shifts in resources without corresponding shifts in accountability/shifts in accountability without corresponding shifts in resources.
	M3.3. Mismatch between resource and productivity expectations.
	M3.4. Lack of clarity for responsibility, expectations, and accountability for performance.
	M3.5. Operational planning frequently pre-empted by emergent crises.
	M3.6. Inappropriate balance between short term operational versus long term strategic focus.
	M3.7. Lack of clarity of operational direction for productive entities (i.e., subsystems).
	M3.8. Difficulty in managing integration of system productive entities (i.e., subsystems).
	M3.9. Slow to anticipate, identify, and respond to environmental shifts.
Metasystem Three Star (M3*): Operational performance	M3*.1. Limited accessibility to data necessary to monitor performance.
	M3*.2. System-level operational performance indicators are absent, limited, or ineffective.
	M3*.3. Absence of monitoring for system and subsystem level performance.
	M3*.4. Lack of analysis for performance variability or emergent deviations from expected performance levels - the meaning of deviations.
	M3*.5. Performance auditing is non-existent, limited in nature, or restricted mainly to troubleshooting emergent issues.
	M3*.6. Periodic examination of system performance largely unorganized and informal in nature.
	M3*.7. Limited system learning based on performance assessments.
Metasystem Two (M2): Information and communications	M2.1. Unresolved coordination issues within the system.
	M2.2. Excess redundancies in the system result in inconsistency and inefficient utilization of resources - including information.
	M2.3. System integration issues stemming from excessive entity isolation or fragmentation.
	M2.4. System conflict stemming from unilateral decisions and actions.
	M2.5. Excessive level of emergent crises - associated with information transmission, communication, and coordination within the system.
	M2.6. Weak or ineffective communications systems among system entities (i.e., subsystems).
	M2.7. Lack of standardized methods (i.e., procedures, tools, and techniques) for routine system level activities.
	M2.8. Overutilization of standardized methods (i.e., procedures, tools, and techniques) where they should be customized.
	M2.9. Overly ad-hoc system coordination versus purposeful design.
	M2.10. Difficulty in accomplishing cross-system functions requiring integration or standardization.
	M2.11. Introduction of uncoordinated system changes resulting in excessive oscillation.

Table 3. Pathologies corresponding to metasystem functions.

Although beyond the scope of this chapter, the metasystem pathologies are also linked to violations of underlying Systems Theory propositions [20, 21]. Pathologies Identification for SoS governance includes assessment across three dimensions of existence, impact, and feasibility. Existence deals with the degree to which the pathology is determined to be present in a SoS metasystem. Impact deals with the degree of severity that the existence of the pathology suggests for the performance of the SoS metasystem. Feasibility addresses the likelihood that, given the current state of the metasystem and its context, the pathology could be addressed with a reasonable chance of successful resolution. Each pathology can be assessed along the three dimensions (Figure 5).

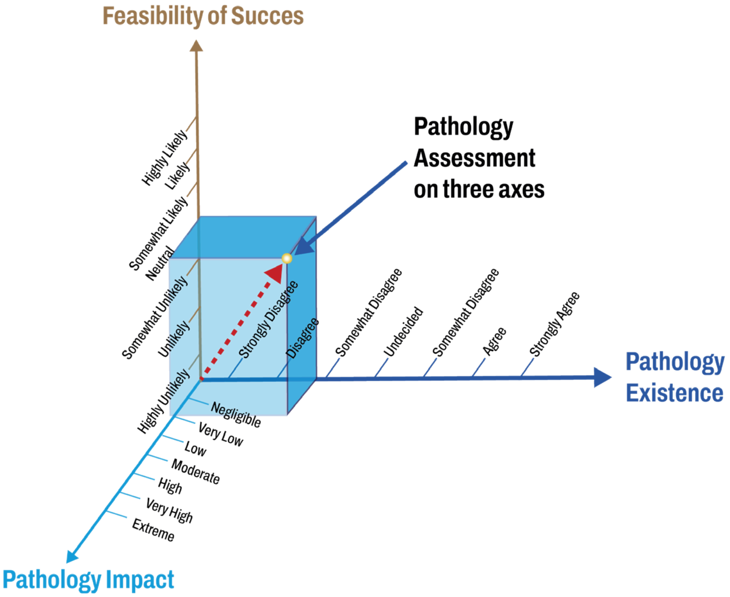


Figure 5. Three-dimensional assessment of pathologies.

4.2. Scenarios for implementation of SoS governance

In this section, we examine three scenarios where SoS governance provides insights and potentially offers different alternatives for governance development. The first scenario examines the prioritization of pathologies for SoS development. This scenario examines the potential for direction, or redirection, of scarce resources based on priority development needs to address pathologies. The second scenario is based on maturing systems-based capabilities for SoS governance. The third scenario examines providing clarity in SoS governance.

4.2.1 Prioritization of scarce resource investment for SoS development

All SoS have resources that are invested to provide for system development and improvement. An example of this is the introduction of a new initiative (e.g., Lean Six Sigma) as something that is recognized as a good thing to do to improve a SoS. However, not recognizing what 'issues' the initiative would address does not mean it is right or a wise investment of scarce resources. SoS governance suggests that scarce resources should be applied to the SoS governance areas that are shown to be lacking (e.g. pathologies of highest existence, impact, and correction feasibility).

Scarce resources should not be squandered on development activities without establishing their specific need and priority contribution to advance the state of SoS governance. Two other contributions of SoS governance might come to fruition in this scenario. First, there is the possibility of 'redirecting' scarce resources, already committed, to higher priority pathologies in need of development. Second, understanding the state of SoS governance provides the opportunity to move beyond piecemeal development to pursue more orchestrated, holistic, and integrated SoS governance development. Instead of looking at development initiatives in isolation, each SoS governance development initiative can be prioritized based on governance needs. Those development initiatives deemed to not have a fit or appear infeasible should be avoided.

4.2.2 Maturation of systems-based capabilities for SoS governance

All SoS have a level of systems maturity. This level is a function of the experiences, activities, design, and execution of the SoS. If this maturity is left to develop on its own, there is no guarantee it will either be at a desirable state or have the desired rapidity of development. Engaging in the purposeful design and implementation of SoS governance contributes to the maturation of systems-based capabilities. Purposeful governance development can follow a cycle of (1) discovery of governance pathologies' existence, impact, and feasibility for resolution, (2) prioritization and ranking of governance pathologies to be addressed with scarce resources, and (3) purposefully engaging high-priority targets for development. The impact of this cycle is the continuing maturation of systems-based capabilities for SoS governance.

4.2.3 Clarity and transparency in SoS governance

SoS governance development can enhance performance in several distinct ways. First, SoS governance makes the SoS and its context clear, explicit, and transparent. SoS governance development calls for the construction of a representation that entails making the SoS governance design explicit. Left at a tacit level, SoS governance can remain ambiguous and potentially create a source of confusion in the execution of governance functions and communication channels. In contrast, by making the SoS governance design explicit, transparency can create clarity in both design and implementation. Second, from a 'baseline' state of SoS governance, the trajectory of governance and the contributions of specific development initiatives can be assessed. This clears the path to continuing, modifying, or eliminating governance development initiatives based on performance assessment. Third, a clear and transparent SoS governance provides for consistency in the current and future trajectory of the SoS. This supports the ability to challenge decisions, actions, and interpretations that uphold a 'status quo' without sufficient explanation, logic, or rationale.

Clarity and transparency are hallmarks of effective SoS governance. The better that SoS governance is understood, the increased likelihood that people will understand their roles, responsibilities, contributions, and accountability for governance functions.

5. GUIDING PRACTICES

This section aims to encapsulate the design and implementation of SoS governance. First, SoS governance is summarized in a concise presentation of key points. Five key points are offered to delineate SoS governance from the perspective of CSG. Second, a set of challenges and cautions are explored for SoS governance implementation. The thrust of these challenges is rooted in our experiences in the implementation of SoS governance.

5.1. A concise explanation of SoS governance for practitioners

SoS governance offers practitioners a new and novel approach to achieve more sophisticated governance. CSG was offered for SoS governance as the "Design, execution, and evolution of the metasystem functions necessary to provide control, communication, coordination, and integration of a complex system" [4, p. 264]. SoS governance can be captured in 5 fundamental themes.

1. ***All systems are subject to the propositions (laws, principles, and concepts) of systems.*** Just as laws govern matter and energy (e.g., the law of gravity), there are propositions that explain and predict the behavior and performance of systems. These system propositions stem from General Systems Theory and are always there, non-negotiable, unbiased, and offer explanations for system performance. Practitioners need to question, *'Do we understand fundamental systems propositions and how they impact the design and performance of governance for our SoS?'*.

2. ***All systems perform essential governance functions that determine system performance.*** Nine system governance functions and 10 communication channels were presented. These functions and communication channels are performed by all systems, regardless of sector, size, or purpose. The functions define 'what' must be accomplished for the governance of a system. Functions are invoked by a set of *implementing mechanisms unique* to a SoS (means of achieving governance functions). For example, a weekly staff meeting and semiannual conferences are examples of mechanisms. Mechanisms determine 'how' governance functions and communications channels are accomplished. Each mechanism can be tacit-explicit, formal-informal, limited-comprehensive, or routine-sporadic in their application. Practitioners must ask, *'Do we understand the mechanisms that perform essential governance functions and communication channels to produce SoS performance?'*.

3. ***Governance functions can experience pathologies (deviations from 'healthy' system conditions) in the performance of functions and communication channels.*** There is no SoS governance system design that achieves perfection in execution. Irrespective of the 'greatness' of a system design, execution relies on too many variabilities to 'assure' complete realization of design intentions. The effectiveness of governance is dependent on the efficacy of identification, assessment, response, and evaluation of inevitable pathologies. Governance supports the achievement of resilience and robustness to withstand and persevere in the

midst of external turbulence and internal system flux. Well-governed systems address pathologies as they occur, while excellent systems continually design out governance pathologies before they escalate into crises. Practitioners must ask, *'Do we purposefully design and continually redesign our SoS to address and preclude pathologies?'*.

4. ***Violations of systems propositions in the performance of governance functions carry consequences.*** System propositions cannot be ignored. Regardless of noble intentions, unconscious ignorance, or willful disregard, violating system propositions carries real consequences for system performance. In the best possible case, violations of systems propositions will only degrade performance. In the worst case, violations of systems propositions will escalate to cause catastrophic consequences or, worse yet, total system collapse. Practitioners must ask, *'Do we understand deficient SoS performance in terms of violations of underlying system propositions affecting functions?'.*

5. ***System performance can be enhanced through the purposeful development of governance functions.*** When SoS fail to meet performance expectations, assessment of contributing deficiencies (pathologies) in governance functions offers novel insights into the deeper sources of failure. SoS performance issues can be traced to issues in governance functions and eventually to violations of underlying system propositions. Through purposeful development, governance can proceed from a more informed position. Practitioners must ask, *'How might our SoS governance functions and communication channels be explored to determine violations of system propositions?'.*

While not a complete set of themes for SoS governance, the provided set captures the essence of the new CSG field for guiding SoS governance design and implementation.

5.2. Challenges and cautions for SoS governance implementation

The design and implementation of SoS governance is not a simple, mundane, or low-resource endeavor. Given the enormity of a SoS governance development undertaking, there are several challenges that practitioners should consider. Our current experience in the application of CSG has provided the following insights for those contemplating such an undertaking for SoS governance development [2]:

1. ***SoS governance development is not the Entry Point:*** As promising as SoS governance might be for advancing system understanding and performance, it is not the highest priority for those who might be considering engagement. Instead, the priority for practitioners is focused on 'their problems'. Thus, focusing on the role of SoS governance in addressing relevant system problems is likely to generate greater engagement.

2. ***SoS governance engagement is not an all-or-nothing endeavor:*** Building on the results of initial explorations of SoS governance and their implications, numerous potential developmental paths can be pursued. Having SoS governance postured as an 'all-or-nothing' endeavor for SoS development is flawed. Instead, the development path might pursue a spectrum of activities (education, training, limited assessment, modeling, etc.) and developmental levels (practitioner, system, project, entity, enterprise, problem) to enhance SoS governance.

3. ***SoS governance is not an 'In-Addition-To' endeavor:*** Unlike more traditional system development activities that seek to address a new concern by introducing a totally new initiative (e.g. Lean, Six Sigma, TQM, CRM, AI Adoption, Internet of Things, etc.), SoS governance functions and communication channels are already being performed by a SoS that is viable (exists). The functions may not be articulated or produce desirable performance, but nevertheless, they are being performed to some degree. SoS governance is focused on understanding and potentially improving what a SoS is already performing. Therefore, the language, thinking, and explorations of SoS governance are applied to an existing SoS where they are already being 'tacitly' performed.

4. ***SoS governance development time and risk should initially fall on the guide:*** It is unrealistic to expect SoS participants to fully engage in a SoS governance development initiative in terms of investment of time and acceptance of 'risk of failure.' Instead, the SoS governance facilitator should bear the initial burden of time investment and risk mitigation as opposed to implementing SoS. This division should continue until the value of investment (time) and utility of SoS governance development combine to produce a risk-value-cost trade-off perceived as being within reason by practitioners. SoS governance development should be conducted in a 'safe to fail' mode, where there are possibilities to take risks without the fear of retribution for falling short.

6. CONCLUSIONS

Designing and implementing SoS governance is not trivial. However, for those practitioners and entities willing to boldly engage in SoS governance development, the potential for enhanced SoS performance is substantial. Four primary points of emphasis exist for this exploration of SoS governance and implementation. First, SoS governance provides direction, oversight, and accountability and was presented as distinctly different from traditional management. Second, SoS governance was presented through a set of 9 metasystem functions and 10 communication channels that produce governance and support continuing system viability. Third, the SoS governance functions are subject to experiencing pathologies as deviations in normal or healthy conditions. Pathologies degrade SoS performance and can eventually lead to catastrophic failure. Pathologies can be assessed for their existence, impact, and the feasibility of successful resolution. Fourth, the contributions of SoS governance include permitting an efficient allocation of scarce development resources, maturing systems-based capabilities, and providing transparency in the purposeful design, execution, and development of SoS governance. Engaging SoS governance development is not a trivial endeavor. The required resources, will, and commitment for SoS governance development are extensive. However, the potential for improvement in SoS performance is significant.

REFERENCES

1. Katina, P. F., Keating, C. B., Bobo, J. A., & Toland, T. S. (2019). A governance perspective for system-of-systems. Systems, 7(4), 54.

2. Keating, C. B. (2022). Complex System Governance. In Complex System Governance: Theory and Practice (pp. 151-186). Cham: Springer International Publishing.

3. Keating, C. B. (2014, June). Governance implications for meeting challenges in the system of systems engineering field. In 2014 9th International Conference on System of Systems Engineering (SOSE) (pp. 154-159). IEEE.

4. Keating, C. B., Katina, P. F., & Bradley, J. M. (2014). Complex system governance: concept, challenges, and emerging research. International Journal of System of Systems Engineering, 5(3), 263-288.

5. Gorod, A., Sauser, B., & Boardman, J. (2008). System-of-systems engineering management: A review of modern history and a path forward. IEEE Systems Journal, 2(4), 484-499.

6. Keating, C., Rogers, R., Unal, R., Dryer, D., Sousa-Poza, A., Safford, R., and Rabadi, G. (2003). System of systems engineering. Engineering Management Journal, 15(3), 36-45.

7. Adams, K. M., Hester, P. T., Bradley, J. M., Meyers, T. J., & Keating, C. B. (2014). Systems theory as the foundation for understanding systems. Systems Engineering, 17(1), 112-123.

8. Whitney, K., Bradley, J. M., Baugh, D. E., & Jr, C. W. C. (2015). Systems theory as a foundation for governance of complex systems. International Journal of System of Systems Engineering, 6(1-2), 15-32.

9. Castelle, K., Bradley, J. M., & Chesterman Jr, C. W. (2022). Systems Theory for Complex System Governance. In Complex System Governance: Theory and Practice (pp. 97-118). Cham: Springer International Publishing.

10. Beer, S. (1979). The heart of Enterprise. Wiley, Cheshire.

11. Beer, S. (1995). Brain of the Firm. John Wiley & Sons.

12. Beer, S. (1985). Diagnosing the system for organizations. John Wiley and Sons.

13. Warfield, J.N. (1976). Societal systems: Planning, policy and complexity. New York, NY: Wiley-Interscience.

14. Keating, C. B. (2022). Complex System Governance. In Complex System Governance: Theory and Practice (pp. 151-186). Cham: Springer International Publishing.

15. Keating, C. B., Katina, P. F., & Bradley, J. M. (2015). Challenges for developing complex system governance.

16. Keating C.B. (2014). Governance implications for meeting challenges in the system of systems engineering field. In: 2014 9th international conference on system of systems engineering (SOSE), IEEE, Adelaide, pp. 154–159.

17. Keating, C. B., & Morin, M. (2001). An approach for systems analysis of patient care operations. JONA: The Journal of Nursing Administration, 31(7/8), 355-363.

18. Keating, C. B., & Katina, P. F. (2012). Prevalence of pathologies in systems of systems. International Journal of System of Systems Engineering, 3(3-4), 243-267.

19. Katina, P. (2022). “Metasystem Pathologies in Complex System Governance” in eds. Keating, C., Katina, P., Chesterman, C., and Pyne, J. Complex System Governance, Springer.

20. Keating, C. B. (2015, May). Complex system governance: Theory to practice challenges for system of systems engineering. In 2015 10th System of Systems Engineering Conference (SoSE) (pp. 226-231). IEEE.

21. Katina, P.F. (2016). Systems theory as a foundation for discovery of pathologies for complex system problem formulation. In A. J. Masys (Ed.), Applications of Systems Thinking and Soft Operations Research in Managing Complexity (pp. 227–267). Geneva, Switzerland: Springer International Publishing.

22. Keating, C. B., & Katina, P. F. (2019). Complex system governance: Concept, utility, and challenges. Systems Research and Behavioral Science, 36(5), 687-705.

23. Keating, C. B. (2022). Complex System Governance. In Complex System Governance: Theory and Practice (pp. 151-186). Cham: Springer International Publishing.

BIOGRAPHIES

DR. POLINPAPILINHO F. KATINA

Dr. Katina is an Associate Professor (tenured) in the Department of Informatics and Engineering Systems at the University of South Carolina Upstate (Spartanburg, South Carolina, USA). He has served different roles in private and public industries, including Design Electric (Charlottesville, Virginia, USA), Politecnico di Milano (Milan, Italy), National Centers for System of Systems Engineering (Norfolk, Virginia, USA), Old Dominion University (Norfolk, Virginia, USA), Syracuse University (Syracuse, New York, USA), and The University of Alabama in Huntsville (Huntsville, Alabama, USA).

Dr. Katina holds a B.S. in Engineering Technology, an M.Eng. in Systems Engineering, and a Ph.D. in Engineering Management and Systems Engineering (Old Dominion University, Norfolk, Virginia, USA). He received additional training at the Politecnico di Milano (Milan, Italy).

His teaching and research interests revolve around topics of Complex System Governance, Critical Infrastructures, Emerging Technologies (e.g., IoT, Smart Grids), Engineering Management, Infranomics, Manufacturing Systems, System of Systems, Systems Engineering, Systems Pathology, Systems Theory, and Systems Thinking.

His profile includes more than 200 peer-reviewed journal articles, conference proceedings, and book chapters. He has also co-authored more than a dozen books, including “Gamification for Resilience” (Wiley, 2023). He has published in over 21 peer-reviewed journals with high impact and visibility. He is a recipient of several awards, including the 2020 IAA Social Sciences Book Award (IAA: International Academy of Astronautics).

Dr. Katina is a Senior Member of IEEE and Epsilon Mu Eta (Engineering Management Honor Society), ABET Program Evaluator, and journal editor for several journals, including Advanced Manufacturing (ELSP), Control and Engineering of Complex Systems Frontiers in Complex Systems (Frontiers Media SA), Cureus Journal of Engineering (Springer/Nature), Discrete Dynamics in Nature and Society (John Wiley), International Journal of Critical Infrastructures (Inderscience), and International Journal of System of Systems Engineering (Inderscience).

DR. CHARLES B. KEATING

Dr. Keating serves as Professor Emeritus of Engineering Management and Systems Engineering at Old Dominion University. His research focuses on Complex System Governance, System of Systems Engineering, Systemic Intervention, and Management Cybernetics. He is a Fellow, Past President, and 2015 Sarchet Award recipient from American Society for Engineering Management for his pioneering efforts in the field. He has published over 170 peer reviewed papers and graduated 32 Ph.D.s. His research has spanned defense, security, aerospace, healthcare, R&D, and automotive industries. He holds a B.S. in Engineering from the United States Military Academy (West Point), a M.A. in Management from Central Michigan University, and a Ph.D. in Engineering Management from Old Dominion University. His memberships include the American Society for Engineering Management, the International Society for the System Sciences, and the International Council on Systems Engineering.

CÉSAR HERAS MENOR DE GASPAR

César Heras Menor de Gaspar is a systems engineer. During his almost 20 years at Isdefe, his career has been focused on engineering activities within the Spanish National Armaments Directorate of the State Secretary for Defence, in the areas of planning for the acquisition of weapons and material resources, development of RPAS (Remotely Piloted Aircraft System) systems and management of defence procurement programmes.

Currently provides technical assistance in the Office of the Next Generation Weapon System (NGWS) Program within the environment of a Future Combat Air System (FCAS), especially in the areas of sixth-generation fighter, remote systems and systems of systems.

Before joining Isdefe, César worked as a systems engineer at HoneyWell, Amper Programs in the avionics and communications systems programs.

In 2021 he was awarded with the Spanish Air Force Medal (Cruz del Mérito Aeronáutico con distintivo blanco) for his work supporting the Spanish National Armaments Directorate

VÍCTOR RAMOS DEL POZO

Víctor Ramos del Pozo is a systems engineer at Isdefe, currently delivering technical assistance for the Spanish National Programme Office for the Next Generation Weapon System (NGWS) within a Future Combat Air System (FCAS), especially in the fields of the Combat Cloud, Collaborative Sensors, Simulation and Systems of Systems. He previously delivered technical assistance for the Spanish National Armaments Directorate of the State Secretary for Defence in the fields of defence industrial base analysis, defence acquisition planning and major defence acquisition programmes management.

Before joining Isdefe, he worked as a systems engineer at EADS-CASA (current Airbus Defence and Space) for the Multi-Role Tanker Transport and Future Strategic Tanker Aircrafts programmes, and at Indra for the Identification Friend or Foe Department.

He is a fellow of the International Council on Systems Engineering (INCOSE) and delivers internal training at Isdefe in the fields of systems engineering and programme management.

In 2015 he was awarded with the Spanish Air Force Medal (Cruz del Mérito Aeronáutico con distintivo blanco) for his work supporting the Spanish National Armaments Directorate.

CHAPTER 6

System of Systems planning and integration

Dr. Michael Sievers, *University of Southern California (michael.sievers@usc.edu)*
Pablo Marticorena San Jose, *Isdefe (pmarticorena@isdefe.es)*

Abstract

The system heterogeneity and dynamic organization of Systems of Systems create opportunities and complexities that necessitate a careful planning and integration process to meet the needs of early users while not constraining future anticipated or unidentified needs. This process involves consideration of and trade-off negotiations involving as many as two dozen factors, accomplished in as many as nine planning activities. This chapter examines the factors affecting SoS planning and the goals of each planning activity.

Keywords

SoS planning, SoS integration, SoS lifecycle, SoS integration

1. INTRODUCTION

This chapter examines the factors and methods employed in planning and integrating the technical aspects of a system of systems (SoS). Notably, the lifetime of a SoS may extend over many years, beginning with modest initial capabilities that continually evolve to meet the needs of new users and usages. A well-conceived planning exercise aims to create a SoS foundation that accommodates near-term connectivity and services for early users while not constraining future anticipated or unidentified needs.

While planning and integrating a SoS shares many activities similar to those used in developing a traditional system, a SoS has unique characteristics that contribute to its utility and complicate its planning and integration [1-5], as discussed in Chapter 1, which occurs in multiple, interdependent activities. The following section discusses planning considerations. Section 3 examines planning for interoperability, Section 4 discusses SoS integration, and lastly Section 5 provides the conclusion.

2. SOS PLANNING CONSIDERATIONS

2.1. SoS performance and behavior features

Before describing planning activities, discussing key features that impact planning is necessary. These features establish the requirements and constraints for SoS architecture, functionality, safety, security, operations, and utility. Table 1 lists key SoS factors and their planning influences.

Factor	Concern	Planning Function
Accessibility	The services and products produced by the SoS are easily accessible to SoS users.	Determine what services and resources clients need and the means and constraints for providing them.
Adaptability	The SoS interactively changes its behaviors and connections to suit individual users.	Determine what adaptability is needed and how to provide it.
Affordability	The degree to which the lifetime cost of the SoS is within anticipated budget constraints.	Establish budgets for development, operations, and maintenance.
Availability	SoS services and resources can be reasonably assumed to be available whenever needed.	Determine the necessary availability of services and resources that meet client expectations.
Composability	SoS components may be connected in many ways, as defined by user goals and requirements.	Architecture planning that assures the composability of services and applications.
Confidentiality	Sensitive data and services are made available only to designated users.	Determine what content, services, and applications are sensitive and the approach to assuring confidentiality.
Disaster management	Preparing for and executing processes and procedures for recovering operations should some or an entire SoS be destroyed by a large-scale disaster such as a fire, flood, or earthquake.	A catastrophic disaster may damage data and computational resources. Determine what must be protected and a plan for providing that protection.
Elasticity	The SoS architecture enables automatic provisioning and release of resources that meet changing workloads.	Planning the SoS architecture to accommodate flexible allocation of resources to match workloads.
Extensibility	The degree to which a SoS can provide for future resources, applications, services, and behaviors.	Plan for a SoS architecture that can seamlessly accommodate future resources and services.
Interoperability	The ability of constituent systems to exchange and use information and services.	Determine how interoperability will be accomplished in new and legacy components.
Maintainability	The degree to which resources in the SoS can be maintained for correcting, preventing, and eliminating faults.	Establish the processes, roles, responsibilities, and audits necessary for evaluating and restoring SoS performance.
Performance	Ensuring that the SoS meets performance requirements under expected usage and environmental conditions.	Evaluate client performance requirements and plan the needed SoS architecture and resources.

Factor	Concern	Planning Function
Quality of service	A measure of the effectiveness and performance of a service or the SoS for performing tasks and meeting user expectations.	Evaluate the quality of service expected by clients and plans for measuring and making repairs or modifications to maintain it.
Real-time	Some SoS applications may require tight time constraints to complete services or deliver products.	Plan for edge or fog computing to accommodate hard real-time applications.
Repairability	The SoS has features that support the diagnosis and repair of failed or malfunctioning resources.	Related to maintainability, plans that make the SoS diagnosable and repairable while preserving required availability.
Resilience	The ability of the SoS to continue providing useful services during disruption and to repair any damage post-disruption.	Plans for incorporating mechanisms that preserve functionality in the presence of resource faults and cyberattacks.
Safety	An SoS that implements safety-critical functions requires special attention to ensure those functions remain controlled and safe.	Plan to add features that protect safety-critical functions.
Scalability	This attribute is related to elasticity; scalability measures how well the SoS can adapt to increasing workloads.	Evaluate and plan for the temporary allocation of resources needed by a client workload.
Security	Protecting the SoS from unauthorized access and ensuring data integrity and confidentiality.	Cybersecurity plans include audits, detection methods, user training, and continual testing. Planning for cyber-resilience involves means for learning and adapting to new attacks.
Spectrum management	Efficient electromagnetic spectrum management in wireless networks to avoid interference and ensure communication.	Plan the use of the electromagnetic spectrum in a way consistent with SoS communication requirements and regulatory provisions.
Trust	Reliance on the SoS' capability, honesty, and reliability.	Determine the extent and means necessary to ensure clients trust interactions with other clients and SoS resources.
Usability	The SoS is user-friendly and easy to operate.	Plan for intuitive and easy-to-use client interfaces, such as web services, RESTful applications, and application programming interfaces.
Users and usages	SoS client needs and expectations drive the need for resources and composability. Increasing numbers of users and threads directly affect the quantity and type of resources and their interoperability.	Plan for changes and growth in client services and resources.
Verifiability	Ensure that the SoS was built correctly and provides the expected capabilities.	Develop plans for performing end-to-end testing of SoS missions, capabilities, and threads.

Table 1: Planning Considerations.

2.2. SoS planning activities

Figure 2 shows the interaction between planning actions that are defined in Table 3. Strategic, tactical, and operational planning are decomposed into their subplans. Lower-level plans tend to have a single purpose, and their details are found in Table 3. Long-term planning extends the goals of strategic planning. Retirement planning is developed from strategic and tactical planning. Lifecycle planning is informed by tactical planning. Each plan may address one or more of the factors in Table 2.

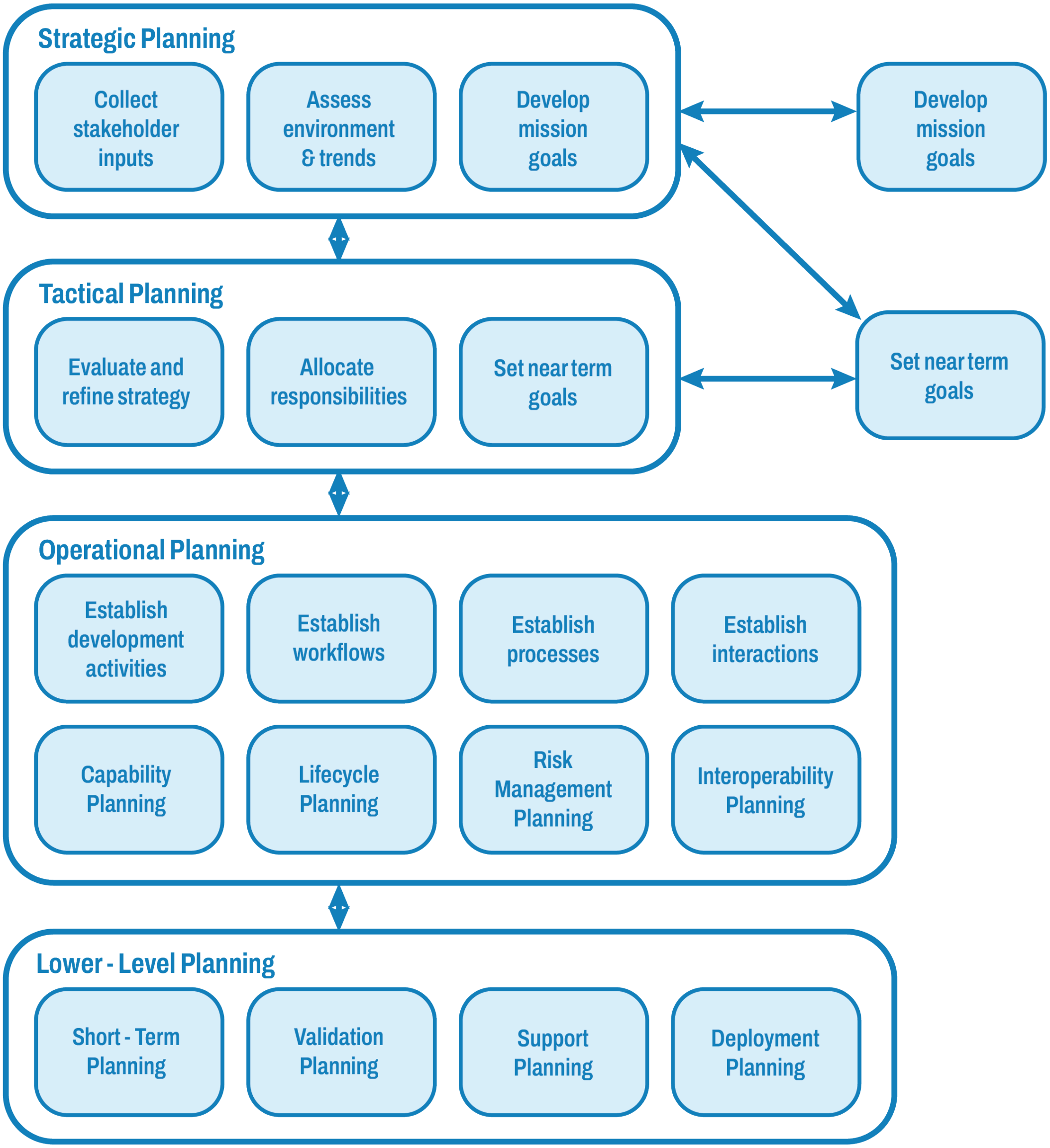


Figure 2: Ontological structure showing SoS concepts and relationships.

<table>
<tr><th>Planning Activity</th><th colspan="2">Purpose</th></tr>
<tr><td><b>Strategic</b></td><td colspan="2">Strategic planning establishes the process for defining an organization’s future direction, needs, and goals. These plans align, motivate, and engage internal and external stakeholders with strategic priorities that reflect mission goals and priorities. A strategic plan is the basis for understanding and accommodating extensibility, adaptability, affordability, availability, resilience, scalability, security, confidentiality, and safety. Strategic plans are often static or minimally changed during the lifetime of a SoS. As strategic plans define the business case and overall missions, they are rarely changed, although capabilities and threads may be changed.
This activity analyzes and assesses the SoS value proposition, missions, trends, best practices, current developments, and gaps; develops a strategy that aligns key stakeholders with missions and the resources needed to achieve them; establishes the processes required to accomplish the strategy; and monitors and evaluates plan progress and replans when necessary.
The plan includes a statement of purpose, analysis of strengths, weaknesses, opportunities, and threats.</td></tr>
<tr><td><b>Tactical</b></td><td colspan="2">Tactical plans map strategic plans into the near-term goals, strategies, and micro-strategies of specific organizations for achieving those goals. These plans include considerations of SoS processing and storage resources, communication, elasticity, maintainability, and interoperability expected of the SoS for carrying out real-time and non-time-critical services and product generation, evaluating product quality, and monitoring SoS performance and resilience. Tactical plans adapt to new needs and realities during the SoS lifetime and are more volatile than strategic plans.
Tactical planning includes creating goals, allocating responsibilities, establishing timelines, determining resources, and assigning tasks.</td></tr>
<tr><td rowspan="5"><b>Operational</b></td><td colspan="2">Operational plans establish the flow of activities that achieve tactical plans. These plans often cover a few months and provide the guidance needed by enterprise managers for operating the SoS. Operation planning comprises the detailed planning of how the SoS will operate in real-world scenarios. This includes defining workflows, processes, and system interactions to ensure smooth operations. This activity initiates Capability-Based, Lifecycle, Risk Management, and interoperability planning.</td></tr>
<tr><td>Capability-Based</td><td>Focuses on identifying and developing the capabilities required by the SoS. This involves defining the desired outcomes and ensuring that the systems within the SoS can achieve these capabilities.</td></tr>
<tr><td>Lifecycle</td><td>Focuses on managing the entire lifecycle of the SoS, from the initial concept through development, deployment, operation, and eventual decommissioning. This ensures that the SoS remains effective and relevant throughout its lifespan. Lifecycle planning primarily takes input from strategic and tactical planning.</td></tr>
<tr><td>Risk Management</td><td>Involves identifying, assessing, and mitigating risks associated with the SoS. This ensures that potential issues are addressed proactively to minimize their impact on the SoS. Risks are often expressed as the likelihood of an unwanted event and the impact of the event if it happens.</td></tr>
<tr><td>Interoperability</td><td>Ensures that the systems within the SoS can work together seamlessly. Plans involve defining standards, protocols, and interfaces facilitating system communication and integration.</td></tr>
<tr><td><b>Long-Term</b></td><td colspan="2">These plans explore SoS options and potential uses many years beyond strategic plans. The primary distinction between long-term and strategic plans is that strategic plans are the means for achieving an organization’s expected future needs. In contrast, long-term plans are a form of unconstrained brainstorming. Strategic plans are on the path that develops SoS requirements and use cases, while long-term plans are unfunded candidates for future goals.</td></tr>
<tr><td><b>Short-Term</b></td><td colspan="2">Short-term plans decide the capabilities available to SoS clients at a point in time. These plans directly influence SoS requirements, personnel assignments, and near-term tasking. An example might be planning the delivery of a specific service used throughout the SoS.</td></tr>
<tr><td><b>Validation</b></td><td colspan="2">Describes the facilities, configuration, resources, test equipment, personnel, test scenarios, and success criteria associated with validating expected SoS functionality and requirements.</td></tr>
<tr><td><b>Deployment</b></td><td colspan="2">Details the plans for how the SoS will be deployed. Deployment options include running the new system alongside an old system, setting up the new system to slowly perform functions of an old system, shutting down an old system and the new system takes over, gradually modifying an old system to the new system, having the new system shadow an old system but is hidden to users, or progressively enabling and testing components of the new system available to users when an old system doesn’t exist. Deployment may also occur with integration.</td></tr>
</table>

Planning Activity	Purpose
Support	Plans the means and staffing to support the operation of the SoS after it is deployed. Support may include service functions that monitor performance, error rates, usage, and outages. Support planning may consist of plans for training operators and users.
Retirement Planning	Describes the plans for retirement and disposal of the SoS. Retiring a SoS does not necessarily imply retiring constituent systems but could remove the framework and utilities associated with SoS interoperability and support. Importantly the plan should address continuation of client services on another SoS. Planning for retirement begins with strategic planning and is detailed in tactical planning.

Table 3: Planning activities details.

3. PLANNING FOR INTEROPERABILITY

An essential aspect of SoS planning involves understanding the form and degree to which services and resources must interoperate [7]. Interoperability can occur at different scales and times and can be applied differently to individual clients and services. Importantly, interoperability must exist with legacy systems that may or may not provide the necessary services.

Commonly, four levels of interoperability may be implemented in a SoS.

- **Foundational interoperability** is the simplest form in which data are shared among systems in an SoS, but the data are not interpreted. Human users interpret exchanged data for use by data clients. It involves establishing communication links and protocols that enable data transmission between systems.
- **Structural or syntactic interoperability** comprises consistent data formats that all systems within the SoS understand. This form of interoperability enables all systems to retrieve, interpret, and process information. It ensures that the data syntax, such as data formats and communication protocols, are standardized so that systems understand and process the data correctly.
- **Semantic interoperability** assures that the systems within an SoS share a common conceptual understanding of data and messages. It also removes the possibility of misinterpreting shared information. It involves defining common data models, vocabularies, and ontologies to ensure that the data exchanged has the same meaning to all participating systems.
- **Organizational interoperability** enables consistent data sharing across an SoS aligned with common goals, needs, expectations, usage, workflows, and enterprise governance.

Thread execution needs and timelines strongly influence SoS interoperability, which impacts SoS architectural decisions, communication options, resilience, maintainability, safety, and availability.

Considerations and accommodations for interoperability are captured in tactical plans as capability statements that describe information sources, uses, and destinations. For example, a mission goal to reduce aircraft accidents is associated with capabilities for collecting heterogeneous aircraft sensor data, collecting weather data, distributing data to situational awareness processors, decision making, and real-time air traffic management control.

Operation and short-term plans describe the technical means for accomplishing interoperability as tactical capabilities dictate. Following the air traffic control example above, interoperability may involve service and message brokers, middleware, data exchange standards and protocols, web services, and proxy servers, among many other options. Real-time traffic control needs information exchanges guaranteed to be free of delays and extended outages.

Interoperability impacts the overall performance of the SoS. Interoperability enhances the adaptability of SoS, allowing it to incorporate new systems and technologies. Planning must be flexible to accommodate changes and upgrades. Also, interoperability ensures seamless integration of constituent systems, which is crucial for effective SoS operation. Planning must consider how to optimize performance through effective interoperability by mitigating or eliminating challenges resulting from:

- Legacy systems may use outdated technologies and communication protocols. Legacy can be challenging to interoperate with newer systems unless significant changes or specialized middleware are used to translate between legacy and newer system syntax and semantics.

- Interoperability requires standardized communication protocols (e.g., HTTP, FTP, SOAP, REST, TCP/IP, UDP, ICMP, etc.) to facilitate information exchange between systems. Planning must define these protocols to ensure smooth interactions. Managing data at scale involves coordinating and consolidating data from multiple sources. A particular challenge is providing interoperability with legacy data stored in siloed databases with obsolete formats that depend on outdated database management software.

- Heterogeneous programming languages, operating systems, and hardware used by different systems may not readily interoperate.

- Systems may use different data formats (e.g., JSON, XML, plain text, CSV, proprietary formats). Translating those formats for interoperability can be difficult and can slow exchanges.

- Communication and processing bottlenecks may slow exchanges, especially as the SoS scales upward. Bottlenecks can worsen when layers of middleware add overhead and additional communication delays.

- Semantic misunderstandings are perhaps the most significant challenge of interoperability because they prevent proper interpretation of exchanged information. Misunderstandings are especially problematic when data from different domains are exchanged.

4. ARCHITECTURE AND INTEGRATION PLANNING

A SoS architecture comprises enterprise resources and their connections as required for providing threads that achieve mission capabilities. Accommodating the attributes noted in Chapter 1 places additional demands on architectural planning. In simple terms, planning the physical architecture of an SoS involves choosing options that efficiently balance opposing attributes. Unfortunately, several realities complicate architecture planning.

Consider a federated architecture to illustrate the complexities in SoS structure and interoperability. A federation is a form of collaborative SoS in which multiple autonomous systems work together to achieve a common objective. Each system within the federation operates independently of the other systems using its data, processes, and control. Collaboration occurs because of well-defined exchange and application invocation interfaces. Systems in loose federations operate almost entirely independently of each other and coordinate through data exchanges and authentication. Tight federations comprise a central authority for coordinating the activities of autonomous systems. Hierarchical federations are structured like tight federations, but the central authority also determines enterprise policies. Peer-to-peer federations have no central authority, and coordination is managed by consensus and predefined protocols. Lastly, hybrid federations are a mix of tightly and loosely federated systems that a central authority may coordinate.

Planning resilience, availability, cybersecurity, maintenance, disaster recovery, and safety accommodations entails knowing and predicting potential disruptions when possible and having procedures to manage the unexpected. Further complications arise when the SoS executes threads that must be completed at time scales that cannot tolerate long outages. The planning process must anticipate the need for robust architectures and develop, test, and revise contingency plans as unexpected fault conditions and novel cyberattacks occur. Commercial cloud providers offer tools, services, and architecture recommendations for building robust and responsive SoS. An architecture plan could include a cost-benefit analysis of engaging a commercial cloud vendor versus building a SoS in-house.

Architecture planning should also consider usages outside mission applications, such as application and service development and deployment, service discovery, maintenance, performance and security monitoring, and auditing. Commercial cloud providers also offer these capabilities. A plan could include evaluating the cost-benefit of using a commercial service provider or building needed capabilities in-house.

Planning a SoS configured as a cloud needs a process for determining the cloud type, e.g., public, private, hybrid, or community:

- **Public cloud** environments use information technology infrastructure and services provided by third-party vendors over the internet. Public cloud resources are shared among multiple organizations and individuals. Key features include:

Scalability	Virtually unlimited scalability to meet varying demands.
Cost Efficiency	Operates on a pay-as-you-go model, reducing upfront costs.
Accessibility	Accessible from anywhere with an internet connection.
Management	The cloud provider manages and maintains the infrastructure.

- **Private clouds** are dedicated to a single organization, providing exclusive access to computing resources. They can be hosted on-premises or by a third-party provider. Key features include:

Security	Enhanced security and control over data and applications.
Customization	Tailored to meet specific organizational needs.
Compliance	Easier to meet regulatory and compliance requirements.
Management	Can be managed internally or outsourced.

- **Hybrid clouds** combine public and private clouds, allowing data and applications to be shared between them. This approach offers flexibility and optimization of existing infrastructure. Key features include:

Agility	Ability to quickly adapt to changing business needs.
Scalability	Leverages the scalability of public clouds while maintaining control over critical data in private clouds.
Cost Optimization	Balances cost and performance by using the most appropriate environment for each workload.
Business Continuity	Enhances resilience by distributing workloads across multiple environments.

- **Community clouds** are shared among multiple organizations with similar requirements, such as regulatory compliance or security needs. The participating organizations or a third party manage them. Key features include

Shared Resources	Cost-effective sharing of resources among community members.
Compliance	Tailored to meet specific regulatory and compliance needs of the community.
Collaboration	Facilitates collaboration and data sharing among organizations with common goals.
Security	Enhanced security measures tailored to the community's needs.

A SoS that executes real-time or time-critical threads may need to consider high-performance, low-latency architectures such as edge or fog architectures. Edge computing reduces latency by physically moving computing and data storage close to the data source or user. Fog computing reduces latency by using geographically distributed computing that extends cloud computing to network edges. A plan should include a process for evaluating architectural options related to the performance of near-term and future mission threads.

Lastly, SoS resources, applications, and services are connected in ways that are consistent with the attributes in Section 1. Other factors [2] impacting SoS integration are shown in Table 4.

Factor	Purpose
Stakeholders	Stakeholders have diverse needs and goals that must be aligned for successful SoS integration. Their requirements drive the design and functionality of the SoS, ensuring that the integrated system meets the collective objectives. Additionally, customers for mission data products, suppliers of information needed to produce mission data products, quality assurance monitors, developers, verification and validation teams, security specialists, accountants, system maintainers, auditors, and managers who need access to specific SoS data, status, and physical resources.
Architecture Development	Besides defining physical and software composition, a plan should identify the process by which a SoS architecture is developed and reviewed, as well as the trades needed to allocate functions and data. The architecture defines the structure and interaction of constituent systems within the SoS. A well-designed architecture facilitates seamless integration, ensuring systems can work together effectively.
Integration Resources	SoS integration sometimes requires special equipment, facilities, and staffing for installation and checkout. Adequate resources, including funding, personnel, and technology, are crucial for integration. Limited resources can hinder integration, leading to delays and suboptimal performance.
Integration Processes	Integration processes comprise the steps to perform integration using integration resources. Initial process descriptions are usually detailed enough to determine the integration resources. Process specifics are developed during operation and short-term planning.
Integration Requirements	Mission functions will have requirements related to accuracy, throughput, capacity, timeliness, safety, etc. Planning should establish how requirements are elicited, validated, tracked, and allocated.

Factor	Purpose
External Influences	External factors such as regulatory changes, technological advancements, and market dynamics can impact SoS integration. These influences must be considered and managed to ensure successful integration.
Risk Management	A risk management plan should be developed for all aspects of the SoS. Regarding integration, risks may be associated with late deliveries, failures and defects, inconsistencies, and assembly errors. A risk management plan provides the process for collecting, bookkeeping, and retiring risks.
V&V	The type of SoS will determine the extent to which verification and validation (V&V) are possible. At minimum, systems should perform V&V on interface protocols to ensure compliance with SoS agreements and standards. A V&V plan explains what is checked and the success criteria. Additionally, planning should indicate V&V roles and responsibilities and the process for documenting and adjudicating V&V failures.
Tailoring and Reuse	Tailoring involves customizing integration processes to fit specific needs, while reusing leverages existing components and processes to save time and resources. Both practices enhance efficiency and reduce integration costs.
Certification and Accreditation	Certification and accreditation ensure that the SoS complies with relevant standards and regulations. This process validates the safety, security, and performance of the SoS, assuring stakeholders.

Table 4: Factors impacting SoS integration.

A plan should define the process for selecting the SoS integration best suited for thread execution as shown in Table 5.

Option	Purpose
Data Integration	Data integration focused on maintaining data consistency across the SoS. Ideally, data should be standardized across all systems. However, middleware or similar services can be employed when system-level standardization is not feasible.
Point-to-Point	Point-to-point integration comprises direct connections between resources. This method is straightforward, but inflexible, leading to complexities as the SoS grows.
Star	Star integration is associated with making direct connections between all resources. It provides a high degree of connectivity but may be difficult to manage.
Hub-and-Spoke	Hub-and-spoke integration has a central hub that connects resources. It simplifies management and reduced the number of connections needed.
Vertical	Vertical integration connects systems within an organization. This approach is suitable within the same organizational boundaries.
Horizontal	Horizontal integration connects systems across organizations. It is ideal for integrating systems that need to collaborate across organizational boundaries.
Middleware	Middleware is glue software that smooths connectivity between software components that may not share common data semantics or syntax. It is useful for integrating heterogeneous systems and data.
SOA	Service-oriented integration connects services to clients through a service broker, hiding transaction details and the client's and the service's location. Universal Description, Discovery, and Integration (UDDI) is a standard for specifying, publishing, and discovering web services.
SOAP	Simple Object Access Protocol (SOAP) is a transport-independent web service messaging protocol that is suitable for scenarios requiring robust security and transaction compliance.
REST	Representational State Transfer (REST) is a simple, scalable, flexible, stateless, client-server communications protocol. REST is ideal for web-based applications requiring lightweight and efficient communication.

Table 5: Thread execution options.

Ideally, the chosen architecture and integration approach should provide capabilities for early users while not constraining future growth. There is a trade between upfront design and implementation costs versus increased lifecycle costs. Creating flexible architectures involves more complicated integration mechanisms with greater upfront costs and risks. However, restrictive architectures that are less costly and risky in the beginning will be more expensive to update in the future.

5. CONCLUSION

This brief chapter overviewed the factors and trade considerations necessary in planning a SoS development. It discussed multiple plan phases consistent with traditional systems engineering. Each plan phase defines processes eliciting and evaluating increasingly finer levels of design and operational detail.

REFERENCES

1. Boardman, J. and Sauser, B, "System of Systems – The Meaning of," Proc. IEEE/SMC Int. Conf. Systems Engineering, Los Angeles, CA, 2006, pp. 118-123

2. Madni, Azad and Sievers, M., "System of Systems Integration: Key Considerations and Challenges," INCOSE Systems Engineering, 17 (3), 2014, pp. 330-347

3. INCOSE Systems of Systems Primer, INCOSE-TP-2018-003-01.0

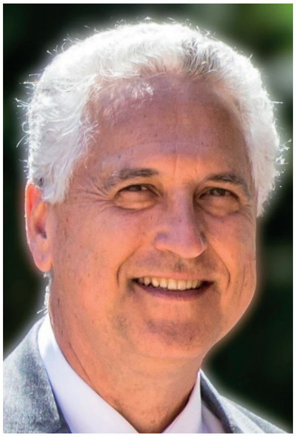
4. Maier, M.W. "Architecting Principles for System of Systems, Systems Engineering, 1 (4) 1998, pp. 267-284

5. Sage, A.P., and C.D. Cuppan. "On the Systems Engineering and Management of Systems of Systems and Federations of Systems," Information, Knowledge, Systems Management, Vol. 2, No. 4, 2001, pp. 325-345

6. Diallo, Saikou & Herencia, Heber & Padilla, José & Tolk, Andreas. (2011). Understanding interoperability," EAIA '11: Proceedings of the 2011 Emerging M&S Applications in Industry and Academia Symposium, Boston, MA, April 3-7, pp. 84-91

BIOGRAPHIES

DR. MICHAEL SIEVERS



Dr. Michael Sievers earned a Ph.D. in Computer Science and MS and BS degrees in Electrical Engineering, all from UCLA. He is currently a senior systems engineer at Caltech's Jet Propulsion Laboratory and on the University of Southern California systems engineering faculty. He specializes in space and ground system mission design, fault-tolerance and resilience, end-to-end information systems, system software, high-performance computing, command and data handling, MBSE, and system trust and reputation. He is currently leading the development of resilience methods for a large U.S. Government system-of-systems and teaches classes in system integration, critical system resilience, and machine learning. Dr. Sievers is an INCOSE Fellow, AIAA Associate Fellow, and an IEEE Senior Member.

PABLO MARTICORENA

Pablo Marticorena holds a BS in Computing Management from the Polytechnic University of Marid and an MS in Computer Engineering from the Universidad Autónoma de Madrid. Throughout his career, he has held various technical and managerial roles, both inside and outside Isdefe, including Data Warehouse administration, SQL/e-CODEX development, digitalization of public administrations, management of technology providers, requirements specification, verification/validation, and RAMS analysis. In his main line of work, he has established himself as an expert in Software Safety and Assurance, accumulating over 22 years of experience in the Air Traffic Management sector. He has supported multiple automation areas, ensuring regulatory compliance of changes made to various critical control systems, in addition to contributing to the integration of methodologies within the life cycles of these systems. He is currently contributing his expertise in requirements engineering to the iTEC SkyNex program, where Isdefe collaborates with ENAIRE and INDRA to integrate technological advancements into Air Traffic Control, creating a unified, efficient, and sustainable ATM environment. Finally, it is worth noting that he remains passionate about his profession and about contributing to the development of Spain and Europe through his technical skills and interpersonal abilities. He also participates as an internal trainer at Isdefe and is an INCOSE CSEP.

CHAPTER 7

System of Systems test and evaluation

Mark Phillips, *University of New South Wales (mark.phillips@unsw.edu.au)*
Dr. Keith Joiner, *University of New South Wales (k.joiner@unsw.edu.au)*
Aurelio Fernández Sáez, *Isdefe (afernandez@isdefe.es)*
Manuel Fernández Astaburuaga, *Isdefe (mfastaburuaga@isdefe.es)*

Abstract

Testing and evaluation (T&E) of a System of Systems (SoS) present unique challenges that differ significantly from those encountered in traditional system-level testing. This chapter presents a comprehensive approach to SoS T&E, emphasizing the need for adaptive frameworks, robust modeling and simulation techniques, and the integration of operational context throughout the test lifecycle. Key considerations include interoperability, dynamic configuration, and the evaluation of performance under realistic, mission-driven scenarios. The chapter also touches on metrics for success, validation strategies across heterogeneous systems, and the importance of stakeholder collaboration. The goal is to ensure that the SoS meets overarching mission requirements while maintaining reliability, scalability, and resilience in complex environments.

Keywords

System, Test, Evaluation, SoS

1. INTRODUCTION

This chapter provides guidance as to how to Test & Evaluate System of Systems (SoS). Test and Evaluation (T&E) of a SoS is a complex and dynamic process that involves multiple stakeholders, integrating multi-proprietary and thus independent yet interdependent systems to achieve a common goal. Unlike traditional system testing, SoS T&E must address emergent behaviors, interoperability challenges, and evolving operational requirements [1, 2].

From the customer's perspective, T&E ensures that the SoS meets mission objectives, operational effectiveness (mission success), and suitability within real-world constraints. Customers, typically government agencies or large enterprises, focus on validating performance, risk mitigation, and ensuring compliance with regulatory and security standards [3]. Conversely, from the contractor's perspective, SoS T&E involves balancing system integration, cost, schedule, and technical feasibility. Contractors are responsible for meeting contract requirements while ensuring that constituent systems function cohesively within the broader SoS architecture. They must navigate evolving customer needs, technological uncertainties, and interoperability constraints across different vendors and legacy systems.

Effective T&E of an SoS requires, to the greatest extent possible, the delineation of predictable performance from whatever emergent phenomena can and will occur through life. This requires a rigorous test methodology, including modeling and simulation, live testing, and iterative assessments throughout the SoS lifetime and the lifecycle of each of its constituent systems [4]. Both customers and contractors must engage in collaborative planning, leveraging agile and adaptive T&E strategies to address the complexities of SoS environments.

2. CONCEPTS THAT AFFECT SOS TEST

The difference in treatment between traditional and SoS T&E can be substantial. Therefore, we should consider the difference when we develop T&E plans. Generally, the constituent systems being integrated have their own levels of maturity in terms of technology, manufacturing, and interoperability readiness levels. This will make it challenging to synchronize the different lifecycles to ensure that the SoS can be tested with a stable functional, configuration, and allocated baselines. This will require close coordination with the customer(s) as part of the SoS governance process. As shown in Table 1, we can identify areas where we should pay close attention due to the integration issues that arise when building a SoS. Note that some of these considerations may also apply to traditional systems.

The importance of managing the overall SoS baseline in coordination with the constituent system baselines cannot be overstated. Failure to achieve this generally results in cost overruns and failure to meet mission objectives.

	Traditional Systems Engineering	System of Systems Engineering	T&E Considerations
Purpose	Development of a single system to meet stakeholder needs and defined performance.	Evolving new SoS capability by leveraging synergies of legacy systems.	Consider the governance model and the requirements for resourcing for the customer(s).
System Architectures	System architecture established early in lifecycle and remains relatively stable.	Dynamic reconfiguration of architecture as the needs change; use of service-oriented architecture approach as enabler.	Define the SoS boundary and provide a T&E Master Plan that pays close attention to the interoperability between constituent systems and the satisficing of the mission of the SoS.
System Interoperability	Defines and implements specific interface requirements to integrate components in system.	Constituent systems can operate independently of SoS in a useful manner. Protocols and standards essential to enable interposable systems.	Develop architecture representations early that identify the interoperability metrics required to assess fitness for purpose. Integrating a SoS can sub-optimize performance overall. Map systems to mission capabilities and identify threshold and objective measures.
Acquisition & Management	Centralized acquisition and management of the system.	Constituent systems separately acquired and continue to be managed as independent systems.	Ensure that the governance model for the SoS is resourced to effectively build and execute a T&E plan. Consider impact to constituent systems and the unforeseen need to account for baseline changes. This implies that it may not be possible to conduct intrusive testing on constituent systems or stop services to perform testing at the edge as the behavior of the SoS will change.

Table 1: T&E Considerations for SoS (adapted from [5]).

3. RISK MANAGEMENT IN SOS TESTING

The T&E of a SoS is an inherently complex and dynamic process that requires a structured yet flexible approach. From the customer's perspective, SoS T&E ensures that mission objectives, security, and operational effectiveness are met. From the contractor's viewpoint, managing cost, schedule, and technical risks is a critical challenge. By employing a mix of modeling and simulation, agile testing, interoperability assessments, and adaptive risk management strategies, stakeholders can enhance the reliability and performance of SoS. As technology advances, future SoS testing methodologies will likely incorporate artificial intelligence, digital engineering, and autonomous testing frameworks to further improve efficiency and effectiveness.

Risk management is integral to SoS T&E, as system interdependencies can introduce unintended failures or mission-critical risks. Both customers and contractors must collaborate on risk identification, mitigation, and contingency planning throughout the T&E process.

3.1. Identifying emergent risks

Emergent behavior is a defining characteristic of SoS, where individual components may function correctly in isolation but cause unintended consequences when integrated. Typically, the constituent systems are already operational and the difference between integrating green field (new) systems and those that are already operationalized means that the behaviors will change, whether positive or negative emergence will be the underlying discovery. Identifying such risks early through system-of-systems hazard analysis (SoSHA) and failure mode and effects analysis (FMEA) is essential [3].

3.2. Stakeholder coordination and communication

SoS risk management requires strong coordination between government agencies, contractors, and third-party vendors. Clear communication channels and shared risk registers help stakeholders to align risk priorities and mitigation strategies [4]. This is more challenging for SoS than traditional systems as the power of each stakeholder will vary as will the lifecycle stage of each constituent system and the programmed resources for remediation and integration. Tools such as Stakeholder analysis can help mitigate problems early through identification of power and influence leading to a more coherent governance model.

3.3. Adaptive risk mitigation strategies

Unlike traditional system testing, SoS risk mitigation strategies must be adaptive and iterative as the constituent systems are rarely aligned in stage of development, upgrade, obsolescence, and capability. The implication of this is that the Test and Evaluation Master Plan (TEMP) should be dynamic in nature. Continuous monitoring, predictive analytics, and rapid-response testing frameworks help detect and address risks as they emerge [2].

3.4. Balancing cost, schedule, and performance risks

Contractors often face challenges in balancing cost constraints, program schedules, and technical performance requirements. Risk-informed decision-making frameworks help prioritize testing efforts, ensuring that critical risks are addressed without exceeding budget or time constraints. SoS have the added issues of competing schedules based on the governance for each constituent system.

3.5. Regulatory and compliance risks

Compliance with government regulations, safety standards, and cybersecurity requirements adds another layer of complexity to SoS T&E. Adhering to Department of Defense (DoD) directives, EU regulation, national regulation as ENS (National Security Framework, by its acronym in Spanish), ISO standards as ISO 27001 and ISO 29119, or other regulatory guidelines is essential to mitigate legal and operational risks [3].

4. CONSIDERATIONS TO SUPPORT SOS T&E

The considerations to support SoS T&E are varied. While they are constantly being addressed by new technologies, the principles of these considerations have been known for over a decade. For example, the following list has been adapted from [6], including several direct quotes in italics.

4.1. Translating capability objectives

For a mission, the capability objectives of the SoS will be stated at a high level; these capabilities drive the goals of the SoS, realized through meeting measures of performance and effectiveness (MOP, MOE). As already mentioned, whereas traditional systems will have technical performance measures (TPM) and key performance parameters (KPP) aligned with well-defined functional requirements, SoS will generally not, due to the integration of constituent systems, many of which will have already been fielded and maybe in the sustainment phase of their lifecycle. Engaging customers in the capability development process and understanding required trades might be helpful.

4.2. Monitoring and assessing changes

SoS are governed differently than traditional systems. The integration of constituent systems, which have governance models established, will make the integration and modification of systems challenging. This requires careful synchronization of changes as each constituent system has its own lifecycle.

4.3. Understanding systems and relationships

Constituent systems are connected through varying degrees of complexity, and understanding their interconnections is essential. A clear grasp of control and information flows within the SoS is necessary. Although these challenges fall primarily within the domain of engineering and development, the associated costs of remediation and integration are important to customers, as they can significantly impact project timelines and resource allocation.

4.4. Developing and evolving system architectures

Identifying and understanding the SoS boundary is essential, along with mapping all interconnections and accurately describing the overall architecture. The SoS is likely to evolve in both design and architecture over time, and this potential for change should be anticipated and addressed during the design phase. Responsibility for these efforts should be shared equally between the development team and the customer’s governance organization.

4.5. Assessment of SoS performance

The MOP and MOE required of the SoS must be clearly defined and well understood. Given that SoS environments are often multi-proprietary and span multiple generations, responsibility for overall design control must be clearly established (particularly regarding measures of supportability (MoS) such as training, maintenance, and spares). It is essential to determine whether the SoS can achieve its mission objectives through the integration of the desired capabilities. All relevant 'ilities' (including interoperability, scalability, flexibility, resilience, adaptability, sustainability, and security) must be explicitly described and revisited with each generational update. These expectations are to be set by the customer.

4.6. Orchestrating upgrades to the SoS

Synchronizing the evolving performance and supportability of all constituent systems must be ensured consistently. Upcoming upgrades to both the constituent systems and the overall SoS need to be identified, along with their potential impact on capability. Effective planning and resourcing of these upgrades are essential. Achieving this requires a strong governance model with alignment among all customers, enabling the contractor to implement the necessary changes efficiently.

4.7. Addressing capability and solution options

The SoS must be aligned to support a defined mission, which directly influences the required MOEs and MOPs. Modifications to constituent systems are likely necessary to ensure successful integration within the SoS. These modifications can be assessed against the relevant 'ilities.' This remains a key area of concern for the customer.

It also implies a need to generate metrics defining the end-to-end SoS capabilities that provide a benchmark for SoS development. Developing these metrics and collecting data (evidence) to assess the state of the SoS is accomplished as part of the SoS system engineered core element assessing the extent to which SoS performance meets capability objectives and hence the mission over time.

4.8. Emergent behavior

Emergent behavior arises from a combination of the behavior and properties of the constituent system elements and structure through allowable interactions between the constituent systems and may be triggered or influenced by a stimulus from the system's environment. Therefore, system engineers need to assess and understand the environment external to the SoS boundary and account for interactions as best as possible. Emergence might not be predicted and may result in a negative impact on the schedule and budget.

T&E Perspective for SoS

Capability objectives are not "specific requirements" assessed through KPPs. Capability objectives are a starting point for developing a statement of expectations at the SoS level and require further specification and elaboration to conduct T&E.

Communication is key, as is understanding how the constituent systems are governed and who has influence and resources. Constituent system changes are often asynchronous to the SoS. Ensure both the capability and technical aspects of the SoS are equally addressed.

The SoS cannot be easily broken apart and tested; in fact it should not, as this will affect the integrated performance. The SoS must be tested in its SoS form, else the behavior can and will change.

Bound the SoS and test as a holistic SoS irrespective of the constituent systems behaviors. Maintain configuration and adject as the capability needs drive changes to the SoS architecture.

Ensure that the MOP and MOE of the SoS are well understood and agreed to by the capability owner and constituent system owners. Can the SoS perform its mission as designed? The measurement may be as much subjective as it is objective!

Changes to the constituent systems can impact the SoS performance. Planning and orchestrating upgrades or updates to the SoS will be essential to testing and will modify the performance and affect behavior.

There may be competing needs between constituent systems and the SoS. SoS capability objectives provide a foundation for identifying systems supporting an SoS, developing an SoS architecture, and recommending changes or additions to systems to meet the capabilities. They also provide the basis for defining and measuring top-level SoS performance and effectiveness.

As emergence cannot be predicted the best course is to allocate budget accordingly. Gaining knowledge from similar integrations can help to inform the scale of impact but not necessarily identify where or how.

5. TEST METHODOLOGIES

Testing and evaluating a SoS requires a departure from traditional system testing approaches due to the complex nature of SoS environments. Unlike single systems which can be tested in isolated and controlled conditions, a SoS is characterized by emergent behaviors, distributed control, and independent lifecycle management of its constituent systems [1, 2]. These SoS phenomena particularly undermines confidence in the following approaches:

- Develop once, maintain through life
- Assuring once thoroughly, maintain through life
- One-off design acceptance testing
- Contracting out development with integration test only once upon delivery
- One-off operational test

Furthermore [7]:

> “Traditionally, systems theory has emphasized the distinction between a system’s internal structure and its external behavior, with the latter derived from the former. This approach has led to a focus on structural representations of systems in cybersecurity protection methods. However, CPS [cyber-physical systems] are heterarchical in nature and consist of multiple, diverse elements that interact both independently and interdependently. As a result, traditional decomposition and predictive methods are inadequate to capture the complexity of CPS. In complex systems, structure and function are intrinsically linked, and a system’s structural characteristics shape its processes and behaviors.”

An illustration of assurance in a heterarchical organization of systems versus a hierarchical organization is shown in Figure 1 to illustrate the different assurance approaches needed.

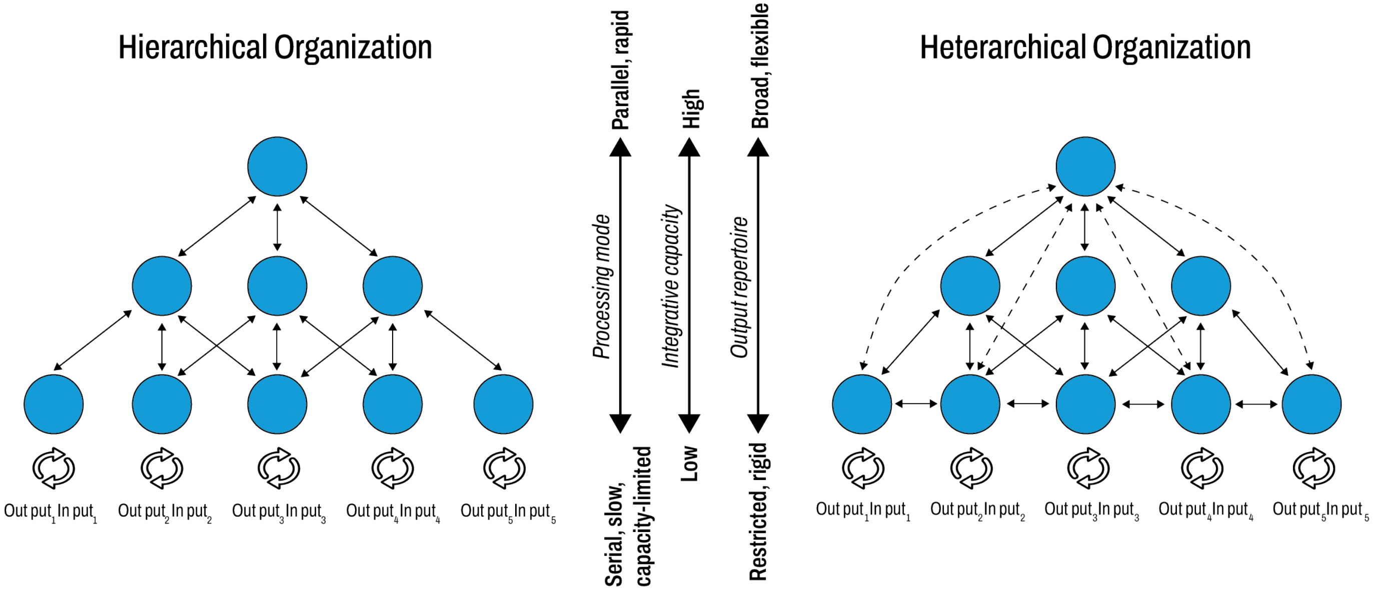


Figure 1: Illustration of different organizational structures in an SoS (adapted from [8]).

Effective SoS testing relies on a combination of methodologies tailored to address these complexities [9, 10, 11].

5.1. Modeling and Simulation (M&S)

M&S is a critical approach in SoS testing, enabling stakeholders to assess interactions, performance, and emergent behaviors before physical testing. Simulations provide a cost-effective way to test various scenarios, stress conditions, and potential system failures. From extensive research analysis, digital twins are defined as 'a virtual representation of a physical system (and its associated environment and processes) that is updated through the exchange of information between the physical and virtual systems' (see Figure 2) [12]. Digital twins and system emulations are increasingly used to analyze performance and predict SoS behavior under different operational conditions.

5.2. Incremental and Agile Testing

Due to the evolving nature of SoS, incremental and agile testing methodologies are often employed, a residual of which is critical throughout life to be *'evergreen'* and thus resilient to emergent behaviors [10,11]. These approaches focus on iterative evaluations, allowing for adjustments based on real-time feedback [3], with generational changes synchronized in a resilience 'battle-rhythm'. Agile testing strategies ensure that changes to individual systems do not compromise overall SoS functionality, particularly in environments where constituent systems are developed by different contractors or agencies who make changes at different times for different customer mixes.

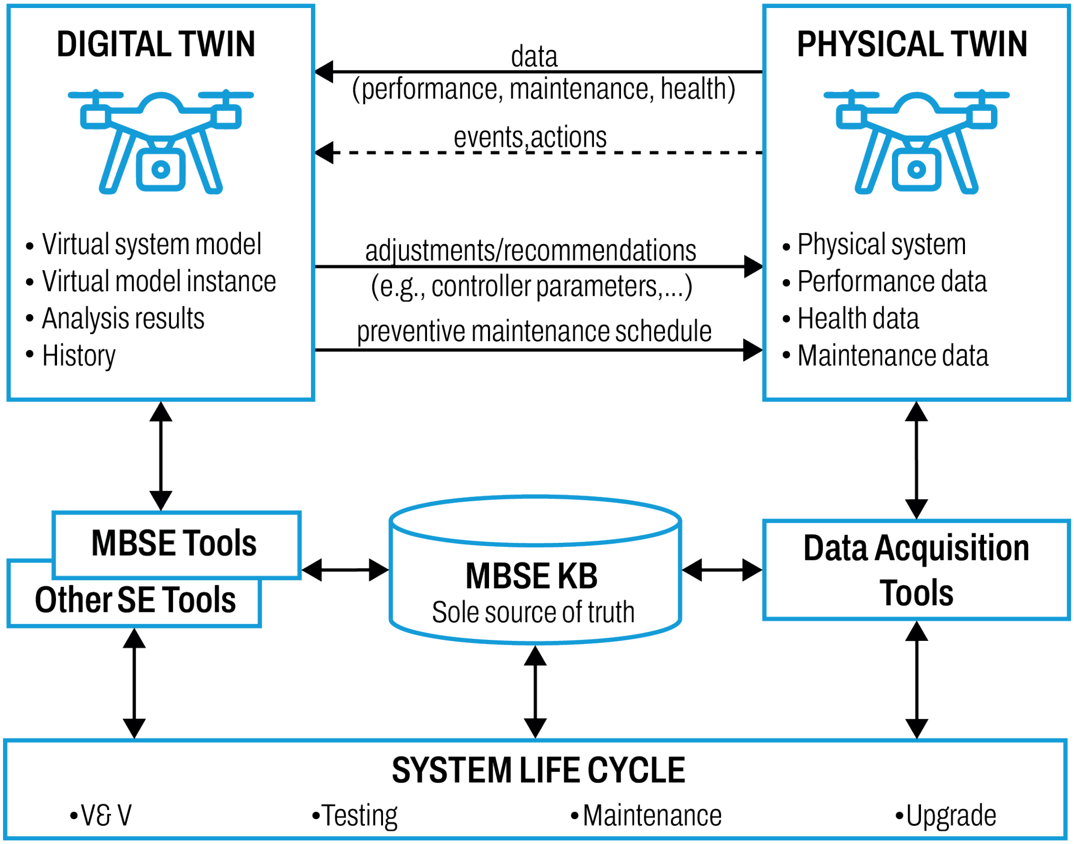


Figure 2: Illustration of a Digital Twin (adapted from [13]).

5.3. Live and operational testing

Live testing in an operational environment is essential to validate the real-world performance of a SoS. Field testing, joint exercises, and war-gaming simulations provide insights into interoperability, resilience, and mission effectiveness. However, live testing is resource-intensive and often constrained by logistical and security considerations [4].

5.4. End-to-End and interoperability testing

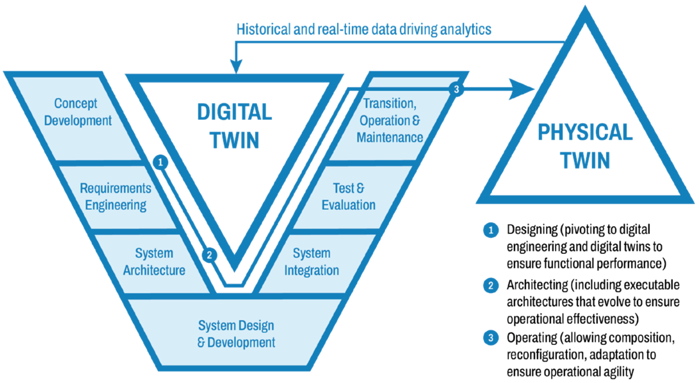
Ensuring seamless interoperability among constituent systems is a primary challenge in SoS T&E. End-to-end testing at the highest level of interoperability is needed (see Fig. 1) to evaluate how well different subsystems communicate, exchange data, and achieve mission success, and through life to deal with emergence. This testing often involves real-time monitoring of interfaces, data exchange standards, and cross-platform compatibility [2]. This aspect of SoS can be particularly challenging for a capability manager, who operates interconnected multi-proprietary systems for overall effect, but has not yet invested in sovereign test-bench representation of the connected systems (i.e., no network integration center (NIC) or live-virtual-constructive (LVC) simulation at the highest level of effect (Fig. 1)) [14, 9].

5.5. Cybersecurity and resilience testing

Given the interconnected nature of SoS, cybersecurity vulnerabilities can have cascading effects across the entire system. Testing for cyber resilience involves penetration testing, threat modeling, and resilience assessments to mitigate potential cyber threats [15, 7]. This requires a thorough and independent analysis of the Cyber Kill chain to identify the attack vectors (mission, attack, variant) across the SoS attack surface.

5.6. Digital twins

Digital twins are particularly effective for the T&E of SoS when progressively developed with the capability and used through life as an SoS T&E basis, as shown in Figure 3 [16]. However, the Model-Virtual-Design-Physical diamond (Figure 4) [17] is perhaps more appropriate, although reused or revisited through life (i.e., helical).



1 Designing (pivoting to digital engineering and digital twins to ensure functional performance)
2 Architecting (including executable architectures that evolve to ensure operational effectiveness)
3 Operating (allowing composition, reconfiguration, adaptation to ensure operational agility)

Figure 3: Illustration of the important through life use of a digital twin for an evergreen SoS T&E basis (adapted from [16]).

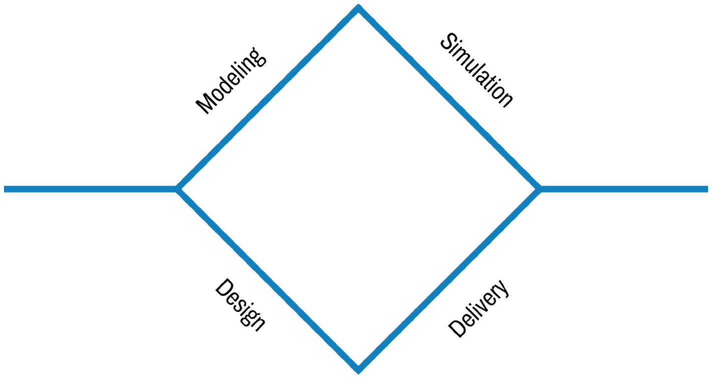


Figure 4: Parallel development of virtual and physical systems (adapted from [17]).

In summary, the T&E of SoS requires a paradigm shift wherever the capability manager and supporting contractors have not yet embraced:

- Continuous Monitoring, Development, Test & Release (which can be enabled by Model-Based Systems Engineering [11], digital twin [12], and digital engineering [18]).
- Highly automated testing.
- Support contracting for a rate-of-change [19, 20, 21, 22, 23].
- Network Integration Centre at the highest level of integration [24].
- Live, virtual & constructive (LVC) simulation for multi-generational interfacing [9].

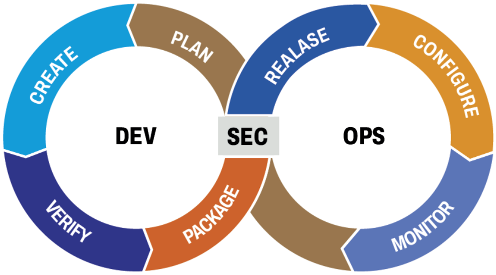
Observed practice (Don’ts)	Desired state (Do`s)	Obstacles
 Defense Acquisition University. June 2010	 CREATE PLAN REALASE CONFIGURE DEV SEC OPS VERIFY PACKAGE MONITOR https://commons.wikimedia.org/wiki/File:Devops-toolchain.svg (modifications licensed CC-BY-SA)	10 U.S.C. $2334 10 U.S.C. $2399 10 U.S.C. $2430 10 .U.S.C. $2433a 10 .U.S.C. $2460 10 U.S.C. $2464 DODI 5000.02 par 5.c.(2) and 5.c.(3)(c)-(d)

Figure 5: Illustration of the need to embrace complexity in management approaches [26].

Put simply, SoS are complex systems where the appropriate management is to create the wherewithal and culture to probe, sense, and respond through life (e.g., Cynefin Framework [25]).

6. GUIDING PRINCIPLES IN SOS T&E

6.1. Embracing continuous development, security, and assurance

One of the key concepts to deal with SoS T&E is to embrace those elements that are complex (ref. Figure 5).

The concepts of DevOps and DevSecOps [27] are key to the T&E needed for SoS:

> *“In the business world, the demand for agility and speed continues to grow. Advancements in technology such as Continuous Engineering, particularly DevOps, allowed some organizations to gain a competitive advantage. However, security concerns have risen because of security breaches, such as massive data breach and leaks, which are forcing organizations worldwide to pay significantly attention to security threats. This is especially true in the context of safety-critical systems, given the possible consequences of security incidents, e.g., loss of life, loss or misuse of sensitive information and major financial loss. In this scenario, high levels of security integration into DevOps are needed. Thus, the need for security to be integrated in DevOps as DevSecOps was first mentioned in 2012.”*

DevSecOps can theoretically lead to a Continuous Authority to Operate (CATO) [28, 10, 29] if there is sufficient automated testing for assurance in these categories:

- Functional/Use (expectation to change).
- Abuse (i.e., safety).
- Misuse/Malicious (i.e., cyberthreat always adapting) [30].
- All non-functional (to the extent possible with fuzz testing [31, 24].

As test cases are added from the top of the list to the bottom there are increasing permutations and a need for greater computation or other test resources. Combinatorial testing helps through coverage of critical factors and functions (multiway) [32, 33]. Hence, combinatorial testing is present in some automated test tools, but to apply these correctly requires additional competencies by testers in combinatorial test design. The process of combinatorial test designing is illustrated in Figure 6.

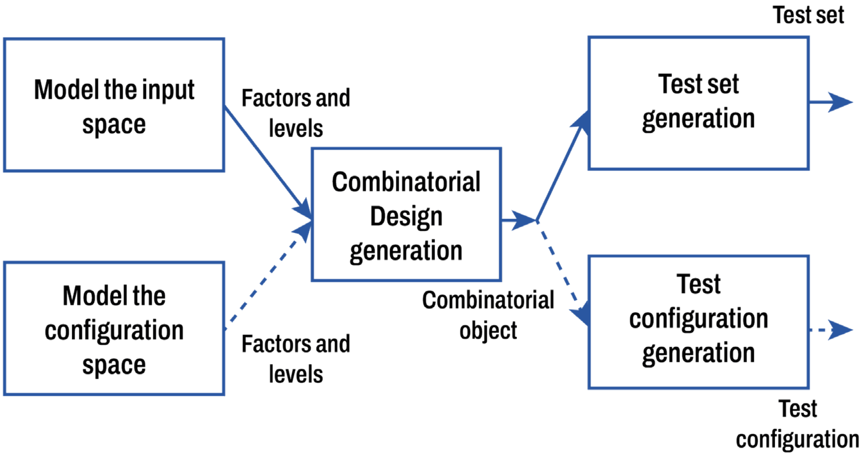


Figure 6: Illustration of combinatorial test design process.

Developing the test infrastructure to realize CATO is challenging. An example is the 10-step methodology for a Simulation, Experimentation, Analytics, and Test (SEAT) Layered Architecture Framework Approaches [34]. Such test approaches are promising, as are improvements in defense and corporate test networks to perform fuzz testing [24] and test methodologies to deal with APIs [35].

"Autonomous and AI-enabled systems present a challenge for integration within the System of Systems (SoS) paradigm. A full system of systems (SoS) testbed is necessary to verify the integrity of a given system and preserve the modularization and accountability of its constituent systems. This integrated system needs to support iterative, continuous testing and development. This need warrants the development of a virtual environment that provides the ground truth in a simulated scenario, interfaces with real-world data, and uses various domain-specific and domain-agnostic simulation systems for development, testing, and evaluation. … Such a virtual and constructive SoS architecture should be independent of the underlying computational infrastructure but must be cloud-enabled for wider integration of AI-enabled software components."

Fuzz-test of all permutations, mixing random and systematic viewpoints seems to work best, especially for critical hardware and software-intensive functions like an aircraft data bus, vehicle LAN, ship IPMS, or rail signaling. An illustration of fuzz testing using a Multi-Armed Bandit algorithm is shown in Figure 7.

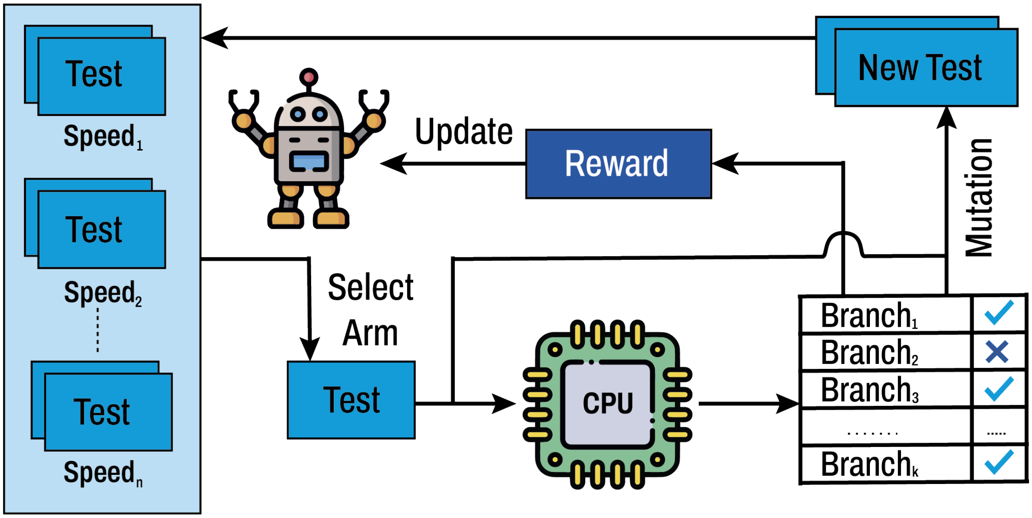


Figure 7: Illustration of Multi-Arm Bandit algorithm fuzz test approach from [37].

6.2. Embracing digitization and digitalization in test

The second major T&E approach for SoS is another general approach from digitalization, Model-Based Systems Engineering (MBSE). MBSE captures the design context, requirements, test metrics, systems design, test cases as the SoS is developed, entering information once to be used many times with substantially reduced documentation, more pervasive and current approval, and the agility to be evergreen (i.e., the authoritative source of truth) [13, 37]. Done properly to include the operational analysis layer, MBSE enables end-to-end scenario or mission engineering to be threaded to test cases and to drive subsequent emergent change management with better targeted regression testing.

MBSE used through life fundamentally helps keep T&E evergreen, and facilitates model-driven test design [38]. Building an MBSE model to cover all relevant SoS elements is challenging [39], as shown in Figure 8. Fundamentally, an MBSE for SoS has to get each proprietary system into a common modelling reference environment, one that is test capable [40], or which allow for transformations between different systems. Since some systems will be pre-digital design, then some retrospective MBSE work is likely necessary to digitize testing.

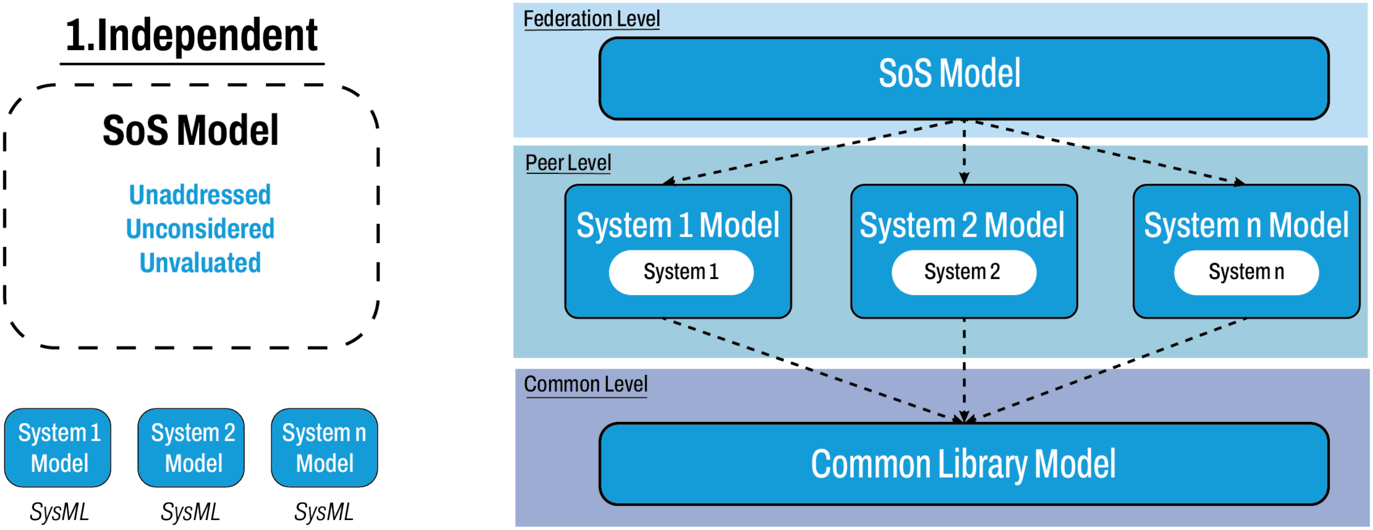


Figure 8: Illustration of MBSE approaches for SoS (adapted from [41]).

7. CONCLUSIONS

SoS T&E requires a different approach to systems testing due to the differences between the two and the properties of SoS. These properties require that the approach taken for T&E considers the impacts of the constituent system baselines, the governance and resourcing for the SoS integration and test, and the difficulties of testing a geographically distributed program that will likely exhibit emergence. This requires close attention to the governance model employed and the anchoring of metrics in mission effectiveness and performance.

A key consideration for SoS T&E that should be kept in mind is that the idea that the SoS can be fully tested before deployment is simply not realistic. It may be more appropriate to view SoS T&E as an evidence-based approach to addressing risk. The SoS systems engineering team identifies issues critical to success of each increment of SoS development, as well as places where changes in the increment might adversely impact user missions and then focuses pre-deployment T&E on them.

Deferring system upgrades until all constituents in an increment are ready to test successfully is impractical and undesirable in most cases. Since most SoS are comprised of already fielded systems, there may not be a discrete fielding decision.

Full SoS level testing can be costly, and it can be very difficult to create test environments which realistically represent the expected results in an operation environment because of the size and complexity of many SoS environments.

REFERENCES

1. Dahmann, J. and D. DeLaurenits (2023). Unique Challenges in System of Systems Analysis, Architecting, and Engineering. Systems Engineering for the Digital Age: Practitioner Perspectives. D. Verma, John Wiley & Sons, Inc.

2. Dahmann, J., Lane, J., & Baldwin, K. (2011). An integrated approach to system of systems effectiveness analysis. The Journal of Defense Modeling and Simulation, 8(1), 3-16.

3. Department of Defense (DoD). (2020). Department of Defense Test and Evaluation Management Guide. Office of the Director, Operational Test and Evaluation.

4. Rebovich, G., & DeLaurentis, D. (2011). Systems of Systems Engineering: Principles and Applications. CRC Press.

5. Vaneman, W. K. (2016). The system of Systems Engineering and Integration “vee” model. 2016 Annual IEEE Systems Conference (SysCon). https://doi.org/10.1109/syscon.2016.7490599

6. Dahmann, J., Lane, J. A., Rebovich, G., & Lowry, R. (2010). Systems of systems test and evaluation challenges. 2010 5th International Conference on System of Systems Engineering, 1–6. https://doi.org/10.1109/sysose.2010.5543979

7. McDermott, T., et al. (2023). Concepts of Trust and Resilience in Cyber-Physical Systems. Systems Engineering for the Digital Age: Practitioner Perspectives. D. Verma, John Wiley & Sons: 473-488.

8. Norman, G. J., et al. (2011). “Current Emotion Research in Psychophysiology: The Neurobiology of Evaluative Bivalence.” Emotion Review 3(3): 349-359.

9. Joiner, K. and M. Tutty (2018). “A tale of two allied defence departments: new assurance initiatives for managing increasing system complexity, interconnectedness and vulnerability.” Australian Journal of Multi-Disciplinary Engineering 14(1): 4-25.

10. Weiss, J. and D. Patt (2022). Software Defines Tactics: Structuring Military Software Acquisitions for Adaptability and Advantage in a Competitive Era. Online, Hudson Institute.

11. Call, D. R. and D. R. Herber (2023). A Case for Model-Based Systems Engineering in an Agile World & Principles for Growth. INCOSE International Symposium, INCOSE.

12. VanDerHorn, E. and S. Mahadevan (2021). “Digital Twin: Generalization, characterization and implementation.” Decision Support Systems 145: 113524.

13. Madni, A. M., et al. (2023). Exploiting Digital Twins in MBSE to Enhance System Modeling and Life Cycle Coverage. Handbook of Model-Based Systems Engineering. C. C. Madni. Cham Springer International Publishing: 527-548.

14. Joiner, K., et al. (2018). Modelling the Efficacy of Assurance Strategies for Better Integration, Interoperability and Information Assurance in Family-of-System-of-Systems Portfolios. International Conference on Complex Systems Design & Management, Springer.

15. Joiner, K. F., et al. (2018). "Four testing types core to informed ICT governance for cyber-resilient systems." International Journal of Advances in Security 11.

16. Tolk, A., Barry, P. and Doskey, S.C. (2022). Using modeling and simulation and artificial intelligence to improve complex adaptive systems engineering. International Journal of Modeling, Simulation, and Scientific Computing, 13 (2) 2241004: 1-19.

17. Richardson (2020). Model Based Systems Engineering: A stepping stone on the path to Digital Engineering. GSFC Systems Engineering Seminar

18. Mukhopadhyay, A., et al. (2023). A Perspective on the Adoption of Digital Engineering Within an Enterprise. Online, Ansys.

19. Cordero, S., et al. (2020). Addressing Obsolescence from day one in the conceptual phase of complex systems as a design constraint. Product Lifecycle Management Enabling Smart X: 17th IFIP WG 5.1 International Conference, Rapperswil, Switzerland, Springer International Publishing.

20. Morgan, M., et al. (2021). "Synergizing model-based systems engineering, modularity, and software container concepts to manage obsolescence." Systems Engineering 24(5): 369-380.

21. Oliver, E., et al. (2022). "A resilience systemic model for assessing critical supply chain disruptions." Systems Engineering(5): 510-533.

22. Donelli, G., et al. (2023). "Concurrent Value-Driven Decision-Making Process for the Aircraft, Supply Chain and Manufacturing Systems Design." Systems 11(12): 578.

23. 23. Herburger, M., et al. (2024). "Building supply chain resilience to cyber risks: a dynamic capabilities perspective." Supply Chain Management: An International Journal 29(7): 28-50.

24. Joiner, K. F. (2024). "Review of Fuzz Testing to find System Vulnerabilities." ITEA Journal of Test and Evaluation 45(4).

25. Daniel, P. A. and C. Daniel (2018). "Complexity, uncertainty and mental models: from a paradigm of regulation to a paradigm of emergence in project management." International Journal of Project Management 36: 184–197.

26. McQuade, M., et al. (2018). Defense Innovation Board Do's and Don'ts for Software (draft). D. I. Board. Online, U.S. DoD.

27. Sánchez-Gordón, M. and R. Colomo-Palacios (2020). Security as Culture: A Systematic Literature Review of DevSecOps. IEEE/ACM 42nd International Conference on Software Engineering Workshops, ICSEW, ACM.

28. Sanders, G., et al. (2021). Integrating ZeroTrust and DEVSECOPS. Online, Carnegie Mellon University.

29. Olena, J. (2023). "Software Standardization and Infrastructure Development Efforts in Support of Unmanned Maritime Vehicle Autonomy." Naval Engineers Journal 135(1): 41-45.

30. Fowler, S., et al. (2025: submitted). "Assessing Cyberworthiness of Complex System Capabilities using the Cyber Evaluation and Management Toolkit (CEMT)." Computers and Security.

31. Brumley, A. (2022). Introduction to Fuzzing Software. ITEA Online (https://itea.memberclicks.net/member-landing-page), International T&E Association: 29.

32. Kuhn, D. R., et al. (2016). Combinatorial Testing for Cybersecurity and Reliability. Information Technology Bulletin, Computer Security Division, Information Technology Laboratory, National Institute of Standards and Technology, U.S. Department of Commerce.

33. Lanus, E., et al. (2021). Combinatorial testing metrics for machine learning. IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW), Online, IEEE.

34. Mittal, S., et al. (2023). "Providing a User Extensible Service-Enabled Multi-Fidelity Hybrid Cloud-Deployable SoS Test and Evaluation (T&E) Infrastructure: Application of Modeling and Simulation (M&S) as a Service (MSaaS)." Information 14(10): 528.

35. Gomez, A. and A. Vesey (2024). On the Design, Development, and Testing of Modern APIs. Online, Software Engineering Institute.

36. Gohil, V., et al. (2024). MABFuzz: Multi-armed bandit algorithms for fuzzing processors. Design, Automation & Test in Europe Conference & Exhibition (DATE), IEEE.

37. Verma, D., Ed. (2023). Systems Engineering for the Digital Age: Practitioner Perspectives., John Wiley & Sons.

38. Alvarado, J. L. J. and T. H. Bradley (2024). "A Case Study-based Assessment of a Model-driven Testing Methodology for Applicability and Metrics of Model Reuse." ITEA Journal 45(4).

39. DeLaurentis, D., et al. MBSE for System-of-Systems. Handbook of Model-Based Systems Engineering. C. C. Madni. Cham, Springer International Publishing: 987–1015.

40. Martell, J. A., et al. "Development of a Digital Engineering Testing Framework for CUBESAT Applications." ITEA Journal 45(3).

41. Swickline, C., et al. (2024). "A methodology for developing SoS architectures using SysML model federation." Systems Engineering 27(2): 368-385.

42. Gorod, A., et al. (2019). Evolving Toolbox for Complex Project Management. Boca Raton, Florida, Auerbach Publications.

43. Dahmann, J., & Baldwin, K. (2008). Understanding the current state of US defense systems of systems and the implications for systems engineering. 18th Annual International Symposium of the International Council on Systems Engineering (INCOSE).

44. Ferreira, F. H., et al. (2021). Reliability in Software-intensive Systems: Challenges, Solutions, and Future Perspectives. 47th Euromicro Conference on Software Engineering and Advanced Applications (SEAA), IEEE.

45. Nachbagauer, A. (2021). “Managing complexity in projects: Extending the Cynefin framework.” Project Leadership and Society 2.

BIOGRAPHIES

MARK PHILLIPS

Mark Phillips is Director of Major Defense Acquisition Programs Strategy at Intel Corporation Government Technologies. Prior to that he was a Technical Fellow and Chief Engineer at Raytheon Technologies focusing on missile technology, and Principal Architect for Lockheed Martin spending the last 2 years there on the National Cyber Range. Mark has over 45 years in defense and government technologies, spanning multiple countries specializing in interoperability between systems. Mark served 20 years in the Australian Army in a variety of roles and transferred to the Royal Australian Air Force reserve to assist in liaison between the US and Australia on training and test areas. Mark is a graduate of the Royal Military College of Australia. holds a Bachelor of electrical Engineering (Honors) from the University of New South Wales, a Master of Engineering in Modelling and Simulation from Old Dominion University, and is a Ph.D. candidate in Systems Engineering at Old Dominion University.

DR. KEITH JOINER

Dr. Keith Joiner, CSC, was an Air Force aeronautical engineer, project manager, and teacher for 30 years before joining the University of New South Wales to teach and research test and evaluation. He is the current Chief Editor for the Journal of Testing by the International Test and Evaluation Association (ITEA) and is also teaching aircraft design. As Defence's Director-General of Test and Evaluation across all service domains, he was awarded a Conspicuous Service Cross, while for doing drawdown plans for the Multi-National Force in Iraq, he was awarded a U.S. Meritorious Service Medal. He is a Certified Practising Engineer and a Certified Practising Project Director, with over 100 published research articles on ensuring the reliability of engineered systems.

His diverse research includes:

1) assuring cyberworthiness, AI-enabled systems, and robotic autonomous systems,

2) using high-throughput test design and complex systems governance, and

3) developing air-sea hybrid vehicles and the electrification of aircraft.

AURELIO FERNÁNDEZ SÁEZ

Aurelio Fernández Sáez is currently Technological Modernization Special Projects Manager at Isdefe, where he leads ICT initiatives for the General State Administration, particularly in the areas of Justice, Employment, Traffic and Social Security. He previously served as Head of the Quality and Systems Testing Area, coordinating strategic projects for the Ministry of Justice and the State Public Employment Service. With over 30 years of experience in Systems Engineering and Digital Transformation, his profile combines strategic vision, technical leadership, and institutional commitment. He holds a degree in Aeronautical Engineering from the Technical University of Madrid and certifications in PMP, Lean Six Sigma and TOGAF. He is a member of INCOSE. He has contributed to process standardization and the management of multidisciplinary teams. He combines his professional activity with a teaching career as a guest lecturer at institutions such as the Technical University of Madrid, the Autonomous University of Barcelona, the University of Málaga, the University of Las Palmas de Gran Canaria, and Isdefe's internal training programs.

MANUEL FERNÁNDEZ ASTABURUAGA

Manuel Fernández Astaburuaga is an aeronautical engineer from the Polytechnic University of Madrid. He is certified in Prince2, ISTQB, COBIT and CSEP for INCOSE. He has 25 years' experience in Systems Engineering, 3 of them in Research and Development on aircraft design at Delft Aerospace University and the following as a consultant in processes and testing in Isdefe. Within them, he was Project Manager of Aena's SW Testing laboratory for 14 years. As a SW testing specialist, he was member of the SSTQB (Spanish Software Testing Qualifications Board) and also member of the Spanish Working Group WP 26 within the SC7 committee of AENOR AEN / CTN 71 "Information Technology" for the SW testing standards definition ISO-IEC 29119. Nowadays he leads a PMO for the IT Directorate in the Spanish Traffic Authority (DGT) and he is Internal trainer in Isdefe on Systems Engineering.

100m
60 km/h
AUTO DRIVING

CHAPTER 8

From system life cycle to SoS evolution

Tom McDermott, *Stevens Institute of Technology (tmcdermo@stevens.edu)*
Miguel Ángel Coll, *Isdefe (macoll@isdefe.es)*

Abstract

Systems of Systems (SoS) do not have rigid lifecycles but rather live in a constant evolution as some constituent elements fade and some new ones join the SoS. This chapter discusses how both the SoS and its constituent elements should be designed and/or architected and managed for evolution. Evolution is driven within the SoS by evolving technologies, processes, products, and tools; and externally by evolving domain drivers such as market competition, economics, and user preferences. This evolution must be managed as both an intentional process and an opportunistic process. It must be led. This chapter discusses strategies for managing SoS evolution based on both SoS management principles and on Innovation System[1] principles. The chapter views SoS evolution as an enterprise leadership challenge and culminates with the qualities of effective SoS leaders.

Keywords

Systems of systems, SoS management, SoS leadership, Innovation systems.

[1] *"Innovation Systems" refers to organized networks of elements (people, businesses, universities, laboratories, governments, technological infrastructure, etc.) that interact to develop, disseminate and apply technological or process innovations.*

1. INTRODUCTION.

By definition, Systems of Systems (SoS) are composed of constituent systems (CS) that are operationally and managerial independent [1]. While this is true, most modern SoS are also dependent on other SoS for sets of capabilities, which blurs the boundaries of the SoS and complicates management. With respect to SoS evolution, changes to the SoS will always occur at the level of the CS [2], which makes SoS change more complicated to manage than in traditional systems. Individual CS can also change independently of the SoS, creating unanticipated effects at the SoS level. Because the SoS provides capabilities that emerge beyond those of any individual CS, an SoS could have unique inputs and outputs beyond those provided by individual CS in their independent uses [3]. In SoS evolution, one must take care that these unique capabilities, and related inputs and outputs, are attained, preserved, and appropriately managed at the SoS level. As a result, SoS evolution must be coordinated at the SoS level and managed across CS and perhaps other SoS.

SoS are becoming more complex. In commercial organizations, most products today are being developed and delivered with connectivity to other systems via commercial networks and software applications. The availability of information is increasing the amount of autonomy and "intelligence" in many systems – even simple products have SoS concerns. Military organizations are also increasing connectivity of battlefields and other assets, while increasing the amount of autonomy in each CS. Military acquisition functions are also encouraging innovation using commercial entities and commercial technologies which reduces their influence at the CS level. All SoS today are emphasizing deployment speed using non-developmental systems, which also leads to reduced CS-level influence. These trends lead to an increase in both the operational and managerial independence of CS, and more complexity for those that must lead SoS evolution. Even since 2019 when the ISO/IEC/IEEE 21840 was published [2], we have been seeing significant changes in the challenges of SoS evolution, to include:

1. SoS composed of larger numbers of CS.
2. Higher operational and managerial independence of CS.
3. Greater connectivity between CS across the SoS, and between SoS.
4. More geographic diversity in CS.
5. More autonomy in CS behaviors, and emerging autonomy in SoS behaviors.
6. More rapid entry of new CS into the SoS.
7. Higher complexity and uncertainty in SoS design and management.

As a result, the leadership and management attributes needed to create and maintain SoS capabilities have changed. This chapter explores approaches to leading and managing SoS evolution, with a focus on innovation and socio-technical change. The chapter begins with a discussion of SoS characteristics that drive evolutionary behaviors. This is followed by a discussion of challenges associated with leading and managing SoS evolution. Methods to guide SoS evolution are then discussed, followed by the leadership competencies necessary for managing SoS.

2. CHARACTERISTICS OF EVOLUTION IN SOS

2.1. Generalities

Evolution in SoS requires a series of methods to structure and manage the set of interacting CS toward a specific set of goals or purposes [4]. A particular pain point in SoS management is the development of methods and tools that help guide, predict, and manage emergent attributes and capabilities [5]. Emergence in SoS is often the result of series of innovations that cause new SoS-level capabilities and attributes to be formed or to become prominent. Thus, one way to view emergence in a SoS is as a series of innovations that, over time, disrupt or transform CS or their operational use in a way that creates new capabilities or attributes that are unique to the SoS. The primary challenge with SoS evolution is to influence these changes toward goals that reflect the needs of SoS-unique stakeholder sets. As such, one component of SoS evolution is to influence the stakeholders in ways that ensure key desirable properties of the SoS are met, which implies that stakeholder collaboration is needed throughout an SoS lifecycle [6].

Most SoS today can be described as *Sociotechnical Systems*[1]. Sociotechnical Systems are technology-driven systems that involve significant human and social participation, and that participation in turn influences the architecture and design of the technical system [1]. Often SoS are also complex adaptive systems, in which both the human/social participation and the engineered system co-adapt over time [7]. The distinguishing

[1] *A "Sociotechnical System" is defined as a system that interdependently integrates social elements (people, organizations, human processes) and technological elements (technologies, infrastructure, software, hardware) to achieve a common purpose.*

feature of a SoS is the behaviors of the "whole" come from individual constituent systems that act independently and autonomously [8].

Systems that have a social interaction demonstrate several consistent patterns. The following characteristics have been selected as most relevant to the concepts of evolution [9]:

- **Self-organization and multi-scale or multi-level hierarchy:** socially driven systems tend to self-organize at lower layers and then create hierarchies as they grow in size, usually driven by series of events. Often the structure of these hierarchies also changes over time. The behaviors that develop at higher layers do not necessarily reflect the behaviors of individual agents or groups. Even *directed SoS* have some level of self-organization, as CS will compete for inclusion in the directed SoS.
- **Autonomy or multi-agent interaction:** individual agents in the system operate autonomously, adapt, and learn as they interact over time. The amount of autonomy in individual agents has been increasing and will continue to increase more rapidly. SoS are formed by formal or informal agreements between autonomous agents, requiring a view of an SoS as a multi-agent enterprise.
- **Emergence:** new behaviors and properties emerge from interactions that are representative of the whole of a system. Emergence arises from the structure of the parts and their interaction, as viewed in a context of interest. These behaviors and properties cannot be predicted from, or reduced to, the properties and behaviors of the constituent parts.
- **Evolutionary development:** goals and objectives, as well as structure and functionality, are in constant change as entities are added, modified, and removed. However, the evolution of the whole happens slowly in comparison to individual agents or components [10] (although it is increasing in speed). Also, goals and objectives in developing a new SoS are very different than in sustaining existing SoS. Methods to evolve SoS level capabilities are very context dependent.
- **Connectivity:** evolution in the system is driven by connectivity, communication (information flow), and collaboration. This applies to both the SoS and the organizations that participate in it. Both the connectivity between CS and the amount of information flow is increasing, while collaboration agreements are becoming more critical. SoS collaboration is an enterprise leadership challenge.
- **Complexity:** the systems are sufficiently large in terms of the number of physical connections, organizational relationships, and information-driven interactions, where they cannot be fully analyzed by conventional (i.e. mathematical) descriptions of system behavior [11]. Qualitative tools and approaches are necessary.

In SoS literature, there is the concept of *evolutionary emergence*, or the study of the evolutionary process the system might take over time, and how to effectively guide that process in the presence of other desirable attributes of the systems (versus undesirable attributes). In this chapter we view this evolution as a set of *innovation pathways*, where groups of innovations become apparent over time at the SoS level based on their disruptive or transformative impact to SoS attributes and capabilities.

2.2. Unprecedented architectures versus incremental evolution

It is important to view SoS evolution as two separate processes: the emergence of a new SoS, which often results in a new or "unprecedented" architecture, versus the sustainment of an existing SoS, which requires a stable foundation. The emergence phase is often competitive, where multiple potential SoS are deploying new technologies and business models. The emergence phase most often follows the *"diffusion of innovation"* model popularized by Rogers, which measures stakeholder adoption [12]. An SoS moves into the sustainment phase when it has been adopted by a sufficient number of users to achieve a stable operating model.

Two forms of evolution are relevant to SoS [13]: those that change the components of constituent systems, and those that change the structure and interactions (system architecture) that are also provided by CS (usually defined as *infrastructure*). Breakthrough or radically new products frequently use new architectures to provide new capabilities. Truly transformative changes most often reflect architectural changes. However, well established architectures are difficult to change due to the implicit knowledge and infrastructure investment they bring. Thus, a further analysis component of SoS evolution must separate architectural decisions from capability decisions. Over time, architectural knowledge becomes embedded in each CS company's organization and procedures. Because this knowledge is now implicit, it can be difficult for any CS to change the architecture of its products. Thus, a SoS that is in its sustainment phase tends to have only incremental changes until it is disrupted by another SoS.

SoS tend to develop and evolve in layers. SoS literature generally describes these as:

1. **Technology layer:** physical aspects of the SoS to include hardware, networking, and other physical infrastructure components. Some of these are not technology, for example air traffic management systems divide the airspace into zones to maintain safety in the architecture.
2. **Applications layer:** the capabilities of the SoS, including software applications, that interact with data and information across CS to provide SoS functionality and user interaction.
3. **Information layer:** the data and information that is exchanged between CS and other SoS that are provided for the SoS to operate effectively. This layer should be focused on interoperability.
4. **Business layer:** the set of relationships across the SoS and CS that respond to changing external drivers and the individual business models of the SoS and its CS.

SoS in the emergence phase are most often driven by technical innovation in the first 3 layers. SoS in the sustainment phase are most often driven by market forces and business models in the business layer, as well as the need for continued interoperability between CS (via standards) that spans the technology and information layers. These are very different dynamics and require very different processes to manage their evolution. The literature across these two phases is strongly divided between *SoS Engineering guidance* and *Innovation System guidance*. SoS engineering guidance assumes we can apply systems engineering and project planning principles to SoS evolution – that they can be "engineered". Innovation System guidance aligns more with social systems and complexity theory and suggests we can only influence SoS evolution, not explicitly plan for it. Practitioners must blend these approaches situationally, based on whether SoS capabilities are emerging or being sustained.

2.3. Planning SoS evolution

The Wave model (Figure 1) is an established framework for evaluating and planning evolution in SoS. It is a top-down framework derived from systems engineering processes. The Wave model recognizes that evolution is continuously driven by input from the external environment (context), and, unlike traditional systems engineering views the analysis of system change is an ongoing process with multiple overlapping increments. The Wave model views evolution as a forward-looking process with feedback at each iteration and attempts to group multiple constituent changes into SoS level architectural changes to create efficiency in the test and validation process [14]. Key aspects of the Wave model are determining a starting point (Initiate SoS), conducting SoS analyses, developing and evolving SoS architecture, and planning and implementing SoS updates. The starting point for an SoS is difficult to define, as the point in time that an SoS is initiated implies that all four architectural layers are sufficiently mature to allow new capabilities to be deployed and adopted (the diffusion of innovation model). The value of the Wave model is the recognition that SoS updates must be planned, and that there is some entity that is analyzing SoS updates and evolving the SoS over time. Except in directed SoS, definitions of the SoS starting point and related analyses can be quite subjective.

Key SoS analysis artifacts include SoS capability-based information – concepts of operation and fundamental constraints; SoS systems information – systems architectural views and CS descriptions; SoS technical information (the technical hierarchy) – performance measures and data: architectural and technical baselines – standards, business rules, connections and interfaces; and SoS management information – contracts and agreements. The successful evolution of an SoS depends on how well SoS leaders communicate artifacts like these, and how well other SoS stakeholders understand them.

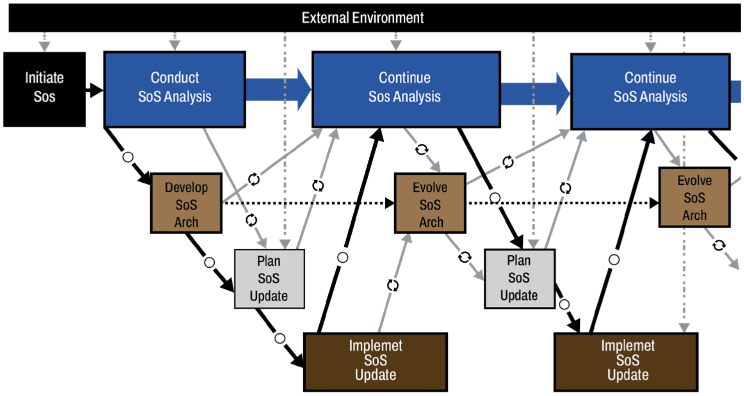


Figure 1. The Wave model (adapted from [14]).

2.4. SoS as an innovation system

Innovation system literature views SoS emergence as "technological transition consisting of major changes in sociotechnical configuration" [15]. The equivalent to SoS wave planning in innovation literature is "transition management" [16]. Innovation system models recognize innovation as a process that spans human and social institutions where lower-level innovations in CS form niches of adoption, which over time produce broader changes in established SoS regimes, eventually resulting in transformation of the existing landscape. Today one can view the public development of driverless vehicle technologies or generative AI transformer models as such an evolution "in-process." The primary aspect of this model is that innovation progresses through social layers and can be modeled as a multi-scale or multi-layer social phenomenon. The transition management literature represents the Wave model as a similar but much messier bottoms-up process of change where the SoS architectures are sociotechnical regimes and the evolutions progress through many components of those regimes (technology, policy, infrastructure, institutions, industries, etc.), enabled by a landscape that is open to change [15]. Figure 2 illustrates this process. One must consider what is happening at any point in time as a snapshot of each layer. Innovations at the CS level, which also introduce new architectures, are a primary driver of regime change and should be a focus of attention in SoS evolution.

The Innovation System of Systems (I-SoS) framework [17] adopts the idea that, in a SoS, multiple innovation systems form relationships. For example, the SoS information, sensing, computing, and control architecture that enables driverless vehicles started as a CS-level challenge (the DARPA Grand Challenges) and emerged mostly in university laboratories. In the driverless vehicle SoS the innovation system providing navigation information (US Air Force Global Positioning System, Google maps, TomTom, etc.) has an interdependent relationship with the innovation system developing automated vehicle control (DARPA vehicle grand challenges, Google Project Chauffer which became Waymo, Tesla, etc.) and the technology companies providing graphics processing units (Nvidia) and low-cost LIDAR[2] (Luminar Technologies). Note that there are multiple innovators including universities, government agencies, very large companies, and smaller technology providers all involved in moving technological niches to new regimes. Many SoS operate within policy constraints that must also participate in the innovation system. In early days of Google's driverless car development, the State of California legislature passed Senate Bill 1298 regulating autonomous vehicles "driving on California roads [18].

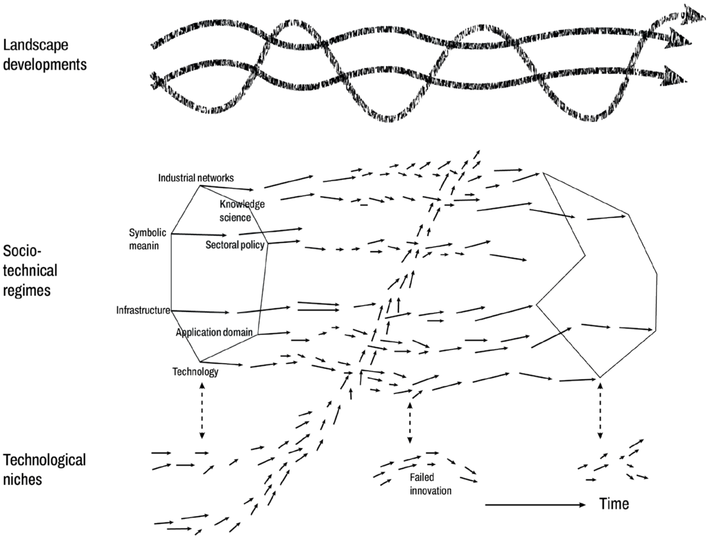


Figure 2. A dynamic multilevel perspective on technology transitions (adapted from [15]).

SoS evolution results from change in three systems: the SoS of interest, the innovation systems, and the broader sociotechnical system where landscape development and regime changes occur [9]. Figure 3 shows the degrees of overlap and separation among the three distinct systems that make up the three systems model. This view has been inspired and adapted from Lawson's "universal mental model" of a system [19]. Each system represents a complex environment of interacting elements, but at potentially different abstraction levels. Changes to the SoS of interest are represented as needs or opportunities, and the SoS of interest also has enablers and barriers that make it easier or more difficult to change. The Innovation System has existing assets and its own enablers and barriers that can be applied in a new system that might be added to the SoS of Interest. These new systems are described as "interventions" in the SoS of Interest since it already has existing structure and capabilities that might encourage or resist new capabilities. The new outputs from the Innovation System must be coupled with the SoS of Interest via integration and evaluation, but that process involves the greater Sociotechnical System that has its own stakeholders, infrastructure, policies,

[2] LIDAR, Light Detection and Ranging is a remote sensing technique that uses laser light pulses to measure distances and create 3D maps. The technology emits laser pulses and measures the time it takes for the reflection to return, allowing the distance to an object or surface to be calculated.

etc. To evolve, SoS stakeholders must embrace and address stakeholders, enablers, and barriers across all three systems. What has value in one system may be at a different abstraction level than the other two. The driverless vehicle example in Figure 4 makes these differences in abstraction levels visually clear.

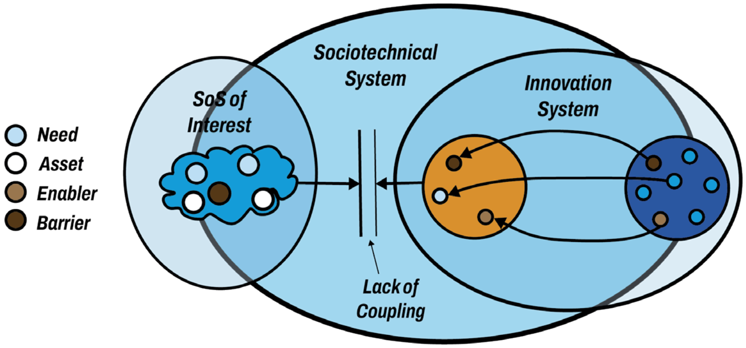
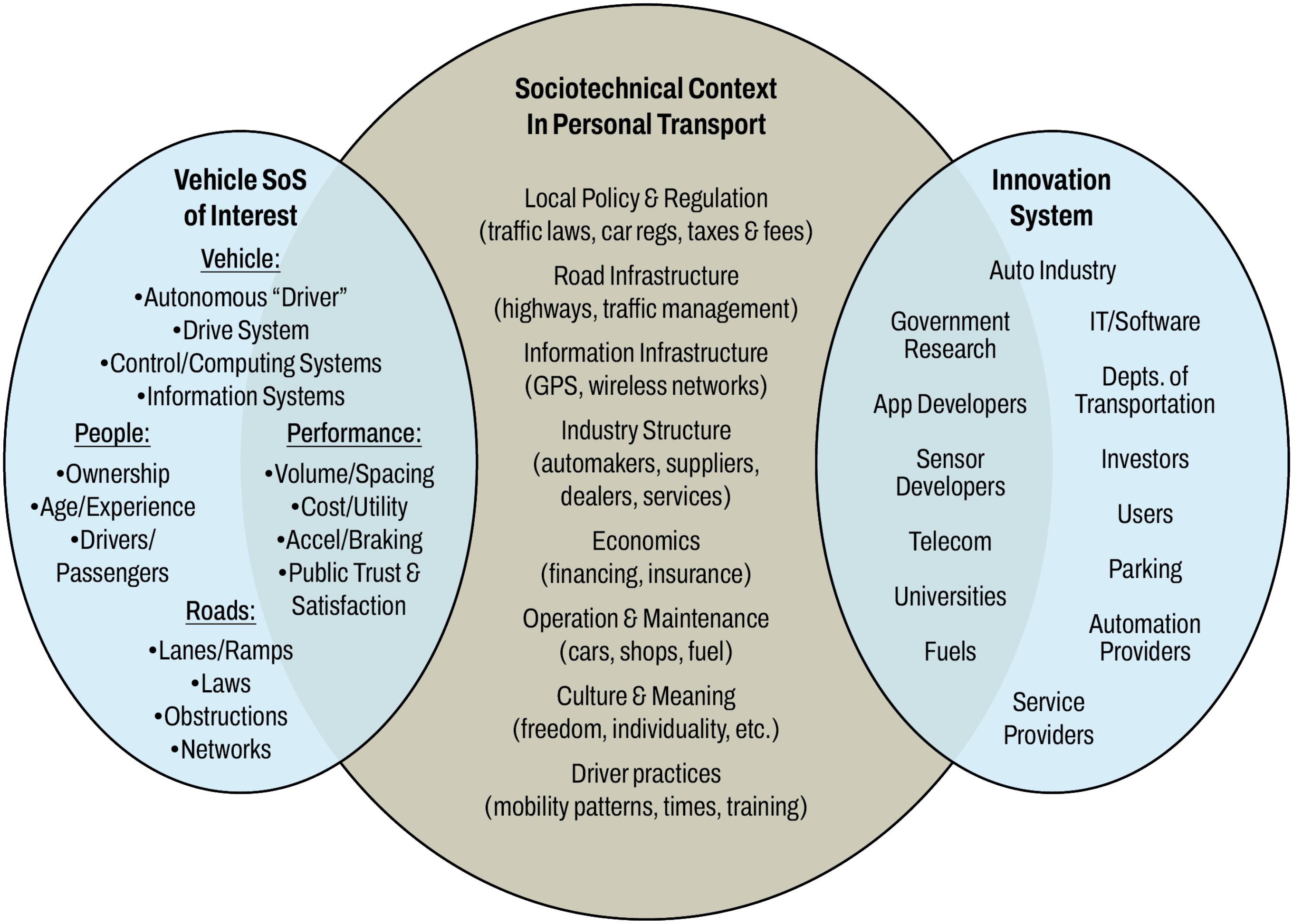


Figure 3. The three systems model (adapted from [19]).



Vehicle SoS of Interest
Vehicle:
•Autonomous "Driver"
•Drive System
•Control/Computing Systems
•Information Systems
People:
•Ownership
•Age/Experience
•Drivers/ Passengers
Performance:
•Volume/Spacing
•Cost/Utility
•Accel/Braking
•Public Trust & Satisfaction
Roads:
•Lanes/Ramps
•Laws
•Obstructions
•Networks
Sociotechnical Context In Personal Transport
Local Policy & Regulation (traffic laws, car regs, taxes & fees)
Road Infrastructure (highways, traffic management)
Information Infrastructure (GPS, wireless networks)
Industry Structure (automakers, suppliers, dealers, services)
Economics (financing, insurance)
Operation & Maintenance (cars, shops, fuel)
Culture & Meaning (freedom, individuality, etc.)
Driver practices (mobility patterns, times, training)
Innovation System
Auto Industry
Government Research
IT/Software
App Developers
Depts. of Transportation
Sensor Developers
Investors
Telecom
Users
Universities
Parking
Fuels
Automation Providers
Service Providers

Figure 4. Three systems model for the evolution of driverless vehicles [9].

The SoS of Interest is primarily composed of well-defined CS and critical underlying system components. The innovation system is a diverse set of stakeholders with highly complex interactions. Both are enabled by the sociotechnical system, which is again formed by diverse stakeholders with complex interactions. Each of the three SoS are comprised of many CS; the actions that take place within each CS define the overall level of SoS complexity and adaptiveness to evolution. Note that some of the CS in an SoS can reside in the sociotechnical system completely independent of the physical manifestation of the SoS but also enabling the SoS to operate effectively. Many of these CS will be associated with policy, laws, and regulations. When evolving the SoS of Interest, a holistic, three system-wide approach is necessary. Many emergent SoS and many changes in SoS sustainment fail because this holistic change leadership is either not present or takes an overly technical view of the SoS.

Most SoS literature discusses methods for SoS management and focuses on the SoS of Interest as the unit of analysis. The SoS of Interest tends to be less complex than the Innovation System that creates SoS evolution and Sociotechnical System that enables or inhibits SoS evolution. Both the Innovation System and Sociotechnical System are complex adaptive systems. While the SoS of Interest might be managed, its evolution in the other two systems must be led. This leadership takes the form of agreement processes – agreements between the interested parties driving the SoS outcomes, the interested parties at the CS level, and the interested parties in the Innovation and Sociotechnical Systems. Some agreements are contracts, but many are informal and operate at lower levels of each participating organization. This leadership also must plan for experimentation and "transition stages" as the SoS is unlikely to evolve all at once. Thus, contracts and other agreements must be flexible in order to account for unplanned effort, cost, and schedule changes in the transition process.

2.5. SoS evolution is change leadership

But who leads? "From a SoS standpoint, innovation is likely to come from a person or group that has good knowledge about the SoS component systems, has knowledge of how to put these component systems together in an overall SoS (or is willing to experiment with the SoS architecture) and is highly motivated by a vision of the potential gains or capabilities the SoS will provide" [20]. SoS evolution is often driven by technical teams who envision and work toward new SoS capabilities but do not have authority to direct SoS changes. This is the case even if the SoS has strong central authorities (is more directed). At the core of SoS evolution, there is a leadership challenge that must be addressed.

The resultant SoS generally possess the characteristics of complex adaptive systems due to the complexity of the organizational aspects of SoS management [21]. To succeed in SoS evolution one must be more focused on the characteristics of the SoS technical organizations than the technical characteristics of the SoS of interest. Many successful SoS have resulted from the vision of a technical leader operating within an organization associated with one or a few CS. For example, Google recruited its initial technical visionaries for self-driving vehicles from the Stanford University team that won the second DARPA Grand Challenge race, who then envisioned and developed the technologies under Google's "Project Chauffer," before spinning the project off to Waymo. Several of the original 16 members of the Stanford team are still involved in Waymo and its evolution of its self-driving taxi service SoS [22].

SoS evolution often comes from decisions at the lowest point in CS evolution that lead to interdependence between individuals who need to work together to realize their vision. Over time these SoS level relationships are formalized through standard ways of doing business. SoS governance should remain separated from management at the CS level, or SoS evolution will become driven by short term reactive changes instead of longer-term evolution [21]. In truth, there is a need for both short term SoS level updates/corrections as well as longer term evolution. In SoS, attempts to drive innovation from the top down will likely be undermined by the lower-level decision-makers in the local CS if it is a different organization. As a complex-adaptive system, the organizations involved must be allowed to self-organize around the dual needs of both the SoS and their local CS. However, as the SoS emergent properties mature the core CS capabilities tend to converge under a single organization through people transfers, mergers, and corporate acquisitions.

2.6. SoS evolution lifecycle considerations

> Abbott states [10]: "Systems of systems evolve in at least three ways. (a) Technology changes, (b) Usage changes, (c) Standards and interfaces change… Systems with these properties do not lend themselves to easy hierarchical control. On the other hand, systems of this sort are not completely formless. Any system, to be useful, must be able to perform specific functions at particular times. Systems of systems achieve this goal in that at any given time (a) they include a collection of (relatively stable) participating systems and (b) they implement a (relatively stable) set of standards and interfaces. But neither the set of participating systems nor the standards and interfaces are fixed forever. They evolve—but slowly."

Because of the complex nature of SoS evolution, there is not a single approach to manage their evolution across the full lifecycle of the SoS. SoS go through periods of emergent change as CS level innovations gain acceptance or CS are retired, and periods of stability where SoS level capabilities are ingrained in regimes and the greater social landscape. SoS stakeholders must take care to recognize and adjust their approach to manage in SoS evolution based on the needs and context for SoS change. Even if the SoS is in a stable period, a CS may undergo change:

- SoS stakeholders may not know when planned maintenance or upgrade of one or more CS occur, or, more importantly, may be unable to synchronize or orchestrate when these changes occur. Awareness is critical.
- An upgrade of a CS may result in deterioration of the SoS architecture because of new obstacles to interoperability. This could include changes to CS that negatively impact CS capabilities needed by the SoS, or changes to standards and interfaces that various CS follow to participate in the SoS. Some level of authority at the SoS level must lead adaptation to the new SoS structure.
- Retirement of a CS or a usage of that CS or a standard does not necessarily imply the retirement of the SoS. Often the replacement of a CS capability that is still needed by the SoS must be paid for by an SoS authority.
- Upgrades of the SoS may not be planned but simply occur as a byproduct of new independent CS being deployed. Again, a level of authority at the SoS level must address integration, test, and training of the user community when these changes occur.

Thus, even when a SoS is in a period of stability, those that have a vested interest in the SoS capabilities and outcomes must expend effort to be aware of pending or even planned changes at the CS level. SoS tend to go through periods where they are more open to new constituent systems and periods where they are more integrated and as a result more stable. More integrated and stable usually equates to "more directed" in terms of SoS authorities. But in truth, almost every highly successful SoS becomes successful in the market because of an unprecedented (new) architecture, which discourages central control in the early lifecycle. The SoS then evolves to be more integrated and under single organizational control, which constrains innovation but creates stability. Concepts from change leadership and from enterprise (business) transformation are more relevant than concepts from management when an SoS is undergoing evolution.

2.7. SoS evolution must be led

While formal standards such as ISO/IEC/IEEE 21839 and 21840 provide details of SoS management processes, there is virtually no literature that discusses approaches to leadership in SoS. Literature searches for content associated with "SoS leadership" returns content primarily on leadership in complex adaptive systems with little direct content on leadership in SoS. However, SoS management is clearly a leadership challenge. ISO/IEC/IEEE 21840 states [2]: "Agreement Processes are crucial for SoS because they establish the modes of developmental and operational control among the organizations responsible for the SoS and the often-independent constituent systems. Constituent systems, which are acquired and managed by different organizations, often hold original objectives that may not align with those of the SoS. Except in the directed SoS case, the SoS organization cannot task a constituent system organization without their cooperation. In an acknowledged or collaborative SoS, these tasks are balanced against the tasks of the CS as a system of interest in its own right. For virtual SoS, agreement processes may be informal, or considered only for analysis purposes." One could ask, "In a context of independent constituents, what are the strategies one can effectively employ to control/steer the evolution of a SoS, especially, in eventual situations of lack of cooperation?" [23]. Literature on leadership of and agreement processes in complex adaptive systems helps to form a model of leadership in SoS, which will be discussed in Section 5.

Thus, the primary challenges to SoS evolution are more associated with leadership than with management. The next section looks at SoS challenges in general and SoS evolution specifically.

3. CHALLENGES TO LEADING EVOLUTION IN SOS

The challenge of SoS is making a SoS evolve towards new outcomes or preserve already desirable outcomes as long as needed, while independent CS may potentially resist change or must evolve toward conflicting CS-level outcomes [23]. Table 1 from ISO/IEC/IEEE 21840:2019 describes at a high level the challenges in SoS evolution as differing from evolution of systems [2].

Clearly from Table 1, while system evolution may succeed based on technical change, SoS evolution can only succeed as an organizational change process. Three references characterize common challenges of evolving SoS from three different perspectives: the SoS of interest [14], CS-level organizational drivers [21], and SoS leadership [23]. The listed challenges are summarized in Table 2 columns 1-3. Column 4 is the authors' summary of the key challenge derived from commonalities between their work.

Systems tend to…	SoS tend to…
Have a clear set of stakeholders.	Have multiple levels of stakeholders with mixed and possibly competing interests.
Have clear objectives and purpose.	Have multiple, and possibly contradictory, objectives and purpose.
Have a clear management structure and clear accountabilities.	Have disparate management structure with no clear accountability.
Have clear operational priorities, with escalation to resolve priorities.	Have multiple, and sometimes different, operational priorities with no clear escalation routes.
Have a single lifecycle.	Have multiple lifecycles with elements being implemented asynchronously.
Have clear ownership with the ability to move resources between elements.	Have multiple owners making individual resourcing decisions.

Table 1. Comparing evolution of systems to evolution of SoS [2].

SoS Leadership Perspective [23]	SoS of Interest Perspective [14]	CS-level Organizational Drivers Perspective [21]	Summary Challenge
Duality of the drivers. CS maintain a separate identity outside of the SoS, pursuing missions of their own. As a result, the SoS will be exposed to influences that would not occur if it were centrally managed. What is locally favorable to CS teams and individuals will drive tasks.	**SoS Authorities.** Each CS has its own local 'owner' with its stakeholders, users, business processes and development approach. As a result, the type of organizational structure assumed for most traditional systems engineering under a single authority responsible for the entire system is absent from most SoS.	**Twin hierarchies are both necessary and useful.** In SoS evolution, role dictates who is responsible, not traditional organizational hierarchy. Alliances of expertise must be established, and the associated task hierarchies must be distinguished from organizational hierarchies.	A strategy must be developed to connect and promote SoS level evolutionary outcomes across CS-level teams. This is via tasks associated with role, not organizational structure. This adds complexity to the management of both CS and SoS. There may not be an entity with authority to define and manage this strategy.
A fractured managerial environment. Along with independence of the CS, this creates barriers for the CS individuals responsible for SoS tasks, resulting in inter-organizational, economic, and sociotechnical disincentives.	**SoS authorities and leadership.** Evolution in a multi-organizational environment is a leadership challenge. The lack of structured control normally present in a CS requires alternatives to provide coherence and direction, such as influence and incentives.	**SoS authority must be earned from the consent of CS developers.** SoS leadership must acknowledge that this consent is not through authority but through shared interest to address SoS needs.	An environment of shared interest for SoS-level outcomes must be created by leadership who may not have positional authority in the CS. This cross-organizational leadership must be clearly defined and individuals incentivized to join in SoS-level outcomes.
Limited to no holistic visibility. The socio-technical barriers between the CS teams create situations where neither SoS-level roles nor the CS level roles can ever assume having full, timely, or permanent visibility on all of the aspects related to the SoS or its full environment.	**Testing, validation, and learning.** CS which are independent of the SoS challenge end-to-end SoS testing. Need a clear understanding of SoS-level expectations and measures, with authority and funding, or it can be very difficult to assess SoS level of performance. CS change cycles may be asynchronous. Full end-to-end testing with every change in every CS would be cost prohibitive. This is often a learning process.	**What is good for me should be good for the organization.** Individuals working across CS and SoS capabilities must be allowed to learn and improve beyond the bounds of their CS responsibilities. This creates loyalty to the SoS outcomes.	Collaboration opportunities between CS teams and individuals must be explicitly created by leaders with either SoS responsibility or accountability, in order to create the visibility of SoS level concerns and outcomes across CS teams. This is a multi-organizational communication challenge.
Limited to no holistic control. No party, even if elected to manage the SoS-level considerations, can assume having any overarching or direct control over the constituents or their evolution processes. SoS-level leadership can only negotiate with rather than control.	**Constituent Systems' perspectives.** Someone must technically address issues which arise from the fact that the systems identified for the SoS may be limited in the degree to which they can support the SoS.	**Managing empty spaces.** CS teams and individuals live within two concentric circles of responsibility, where the inner circle is the core of their responsibility and the outer is the limit of their authority. The in-between is the area of discretion where SoS capabilities are evolved.	SoS responsibilities and accountability must be negotiated formally into CS-level contracts or informally into shared business models. The formal and informal negotiated relationships may be very complex, especially if there is no central SoS authority.
More potential for incompatibility and conflicts. When a responsible party in one CS sees a SoS, a responsible party from a different CS may be seeing a different SoS or no SoS at all. This could be pure lack of visibility or bias. It is difficult to introduce evolutionary change without causing inconsistencies.	**Capabilities and requirements.** SoS needs evolve independently of individual CS needs, often requiring CS to take on new requirements or to replace participating CS with other participating CS. These relationships must be formally documented, through standards, requirements, etc. These may not all be known up front.	**CS responsibilities.** These CS developers must "sign their work," they must account formally for their responsibilities to complete work necessary to the SoS. This must be communicated to other CS teams to manage inconsistencies.	In practice, some SoS-level accountable structure must be in place to oversee (if not govern) individual CS responsibilities and create documented communication of agreements. This may not be present in initial SoS development but must evolve as stability of the SoS becomes normalized. This often occurs by moving CS under SoS common authority.

Table 2. Challenges in SoS evolution.

4. METHODS TO GUIDE EVOLUTION IN SOS

For engineered systems, there is some authority that serves to advance the emergent goals of the SoS, and also a set of stakeholders who have task-oriented responsibility at both the CS and SoS level to create, manage, and sustain the SoS in its states of evolution. With SoS, authorities tend to evolve across the lifecycle but those responsible for tasks tend to persist. The Wave model views this as a systems engineering process where intentional analysis and planning of the next state of evolution can be done. The transition management model views this as an innovation process where these states of evolution must be guided but cannot be discretely planned. In practice, both models are relevant. Based on the summary challenges in Table 1, analysis of the next state of evolution must reflect not only how this evolution will occur, but also who will enable and create the changes.

4.1. Organizing for SoS evolution

The RACI model [26] is a useful organizational framework for clarity of agreements in SoS evolution and can be used to classify where SoS leadership can be most effectively applied. The RACI acronym stands for Responsible, Accountable, Consulted, and Informed. SoS evolution actually occurs from individuals or teams in the "responsible" category (those who complete tasks necessary for SoS evolution), not the "accountable" category (the authority, or the one person ultimately responsible for SoS outcomes). In some SoS, there is no one accountable authority. The "consulted" are those whose opinions are sought in order to aid evolution, often subject matter experts, with whom the responsible and accountable must engage in 2-way communication. This collaboration process must be enabled between CS responsible and accountable stakeholders. Finally, the "informed" are those that need to be kept up to date on evolution, particularly task completion. One might consider this as a way that the details of current SoS state are transferred across all the CS. Even this can be difficult in a SoS as the individual CS may have competitive reasons to protect the details of their current state [25].

Applying RACI to the summary challenges of Table 2, we can suggest the practices in Table 3:

Summary Challenge	Change Practice in SoS Evolution
1. A strategy must be developed to connect and promote SoS level evolutionary outcomes across CS-level teams. This is via tasks associated with role, not organizational structure. This adds complexity to the management of both CS and SoS. There may not be an entity with authority to define and manage this strategy.	There might not be a central authority for SoS evolution. There will be one or more leaders who are accountable for SoS outcomes and may have variable levels of control or influence over CS-level tasks that lead to CS-level evolution. The strategies to create and maintain SoS capabilities and outcomes will evolve with SoS evolution, and the accountable leaders may evolve with them. The direction of SoS evolution must be informed by a strategy. This can be thought of as an enterprise transformation process [26].
2. An environment of shared interest for SoS-level outcomes must be created by leadership who may not have positional authority in the CS. This cross-organizational leadership must be clearly defined and individuals incentivized to join in SoS-level outcomes.	SoS must be built on shared interests and/or concerns. Efforts from a stakeholder who has a leadership position must be spent to establish and maintain shared interest. Leaders must find actionable shared interest between CS teams and communicate it across a large and possibly remote set of responsible teams [23].
3. Collaboration opportunities between CS teams and individuals must be explicitly created by leaders with either SoS responsibility or accountability, in order to create the visibility of SoS level concerns and outcomes across CS teams. This is a multi-organizational communication challenge.	A person who has a leadership position with respect to the SoS, which can be either responsible or accountable, must create an environment where CS-level relationships can form and where responsible task owners can collaborate and become interdependent with respect to their task responsibilities. This interdependence serves to distribute power across CS and must be an active process when SoS capabilities are being updated or evolved. This includes engaging with consultants on behalf of the CS and keeping other stakeholders informed.
4. SoS responsibilities and accountability must be negotiated formally into CS-level contracts or informally into shared business models. The formal and informal negotiated relationships may be very complex, especially if there is no central SoS authority.	Shared interests are necessary but insufficient to succeed at SoS work tasks when CS level priorities interfere with SoS work responsibilities. Incentives for SoS level work must be applied and captured into documented agreements that are negotiated between CS, whether formal contracts or other more indirect business relationships. These incentives may be economic, partnership-based, defined in policy, etc. [23].

Summary Challenge	Change Practice in SoS Evolution
5. In practice, some SoS-level accountable structure must be in place to oversee (if not govern) individual CS responsibilities and create documented communication of agreements. This may not be present in initial SoS development but must evolve as stability of the SoS becomes normalized. This often occurs by moving CS under SoS common authority.	This is the most difficult practice to define, as there are many means to create accountability for SoS capabilities and outcomes. The most straightforward is to become more directed, using explicit contracts with CS for SoS requirements. The danger with this approach is that these contracts are difficult to change as SoS capabilities emerge and their value are learned, suggesting these contracts should also evolve and be flexible in early stages of the SoS. The trend in commercial SoS is for the organization that gains the most value from the SoS to gradually grow its authority by moving CS under its authority, often by mergers and acquisitions. This is a more evolutionary process. There is also opportunity to harness network effects, which are often formed around policy, standards, or common economic interest (such as regulations) [23].

Table 3. SoS evolution challenges and associated change practices.

Table 3 emphasizes that analysis of SoS stakeholder relationships is often more important than SoS capability relationships when planning SoS evolution. Practices to evolve these stakeholder relationships are not well covered in existing SoS literature but can be derived from innovation systems literature and enterprise systems literature. SoS leaders are accountable for SoS evolution but only sometimes have authority to enforce change. Summarizing Table 3 from a leadership perspective, SoS leaders must expend efforts to:

- Define and communicate strategies for SoS evolution.
- Define and create actionable areas of shared interest across multiple CS teams.
- Create an environment where CS-level relationships can form and where responsible task owners can collaborate.
- Define and apply incentives for SoS-level work and negotiate these into CS agreements.
- Evolve business models and relationships that make CS accountable for SoS-level outcomes.

At a high level, these change practices imply those that are responsible for tasks are:

- Aware of the structure of the SoS and the associated context or external drivers, as well as strategies for SoS evolution.
- Know the predominant stakeholders across at least the CS level teams where there are interdependencies, with whom they might share interest.
- Know sufficiently the SoS-level stakeholders that should be consulted with and kept informed.
- Respond to SoS-level change goals and strategies with defined work tasks that are negotiated into agreements.
- Accept responsibility for implementation (sign their work).

Generally, for those that are responsible, these are technical interests and concerns. Those that are accountable (even if they lack authority) must be concerned with the SoS as an enterprise with technical, business, and other mission goals. These include sociotechnical concerns. Those who are consulted have an important role in creating knowledge across all the sociotechnical dimensions of the SoS.

4.2. Preparing for SoS evolution

The value of the Wave model, at least as a strategy, is it defines "SoS analysis" as an active ongoing process at the SoS level. Based on the I-SoS framework and additional research on enterprise transformation [26], we recommend those that are accountable to SoS-level change goals regularly conduct analysis of SoS evolution in four dimensions: SoS definition, SoS actors, SoS change goals, and SoS implementation.

1. **SoS definition:** When analyzing SoS evolution paths, one should start with clearly defining the current state of the SoS of interest, including the context (or current environment) that constitutes the sociotechnical context the current SoS is operating in. Note from Figure 3 this includes the stakeholder needs it satisfies, the current set of assets it uses (likely not just a list of CS but also the details of how each contribute to SoS), and the enablers and barriers to SoS change (a very important aspect that is often neglected).

2. **SoS actors:** Those accountable for SoS change goals must define all of the actors at each CS/SoS layer and "what they bring with them" – their business models, resources, networks, institutions, etc. This should also include knowledge if available about what other programs are driving CS level change.
3. **SoS change goals:** This should capture the desired end state of the SoS evolution in terms of new desired capabilities, who would use them, and what the priority of each is. The goal of this analysis is to identify primary performance measures of the current SoS and of the desired evolution, and who would be tasked to achieve these.
4. **SoS implementation:** The implementation dimension serves to build a model of all dimensions considering new SoS outcomes (or goals), and the interactions that cause them. It is likely that the desired end state will not happen all at once, so dividing these into transition periods will help the planning process for the evolution. The implementation dimension should include estimates of schedule and cost, and where the resources might come from. Note from Figure 3 we call this an "intervention" into the current SoS and the implementation dimension must use resources from the innovation system to create the desired changes in the SoS.

It is important to clearly define what is emergent in an SoS independent of its CS. We cannot usually predict exactly what emerges, but one can look at change trends and events that are precipitating change as signals of how new emergent capabilities should occur in the SoS. For example, what emerges in the SoS context related to self-driving vehicles, independent of vehicles as systems? In one Waymo business case, it is a new kind of taxi service that operates without taxi drivers. What are the independent CS that make up a self-driving taxi service? What are their relative cost, performance, availability, user interfaces, etc.? Who are the independent organizations that contribute to this SoS? How has it evolved?

To plan for SoS evolution, the SoS accountable leads should assess [27]:

- What is the current state of the SoS? (The SoS of Interest per Figure 3).
- What are the new desired SoS capabilities? (Figure 3 describes these as the Interventions).
- What is the priority of each desired capability?
- What resources exist that we can work with? (This is the resources in the Innovation System).
- When do we want to introduce these capabilities?
- What will be the resulting business model?

The desired analysis framework that captures all stakeholder intentions would then represent all components of the related SoS', distinguished by societal layer and by an interrelationship model that considers enablers and barriers, actors, interactions, and outcomes of the desired SoS evolution. This interrelationship model can be mapped and visualized as a tool to gain stakeholder consensus on SoS structure and performance goals, and to further computationally model the system [28, 29]. We will propose two qualitative tool frameworks in the next section. These are both holistic tools, since [10]: "a system of systems is best viewed not as a hierarchy built of component systems but as an environment within which other systems operate and which can support the addition of new systems that build on systems already in the environment. Furthermore, to fully understand a system of systems not only must it be viewed as an environment for other systems, but it must also be understood in terms of the larger environment within which it and its participating systems exist. In other words, a system of systems perspective requires one to look outward from a system rather than inwards towards the system's hierarchical components."

4.3. Two tools for analyzing the paths of SoS evolution

Returning to the three systems model, most systems analyses separate or avoid the complexity of these three systems all together when designing an intervention or evaluating a project's feasibility. Having a toolset that supports a SoS approach, bringing the three systems together for integrated analysis, provides a novel approach in the field of systems of systems engineering and complex adaptive systems. A toolset that is designed to consider these three systems in their entirety provides analysis of the interactions, relationships, and evolution between and within these systems to offer an unparalleled vantage point from which to gauge the potential for innovation-fueled impact [8].

SoS analysis is a systems thinking challenge. The tools at least initially are qualitative and designed to encourage holistic thinking about the three systems at play. In traditional social systems analysis, we are encouraged to view an SoS at three levels: the macro (societal, institutional), meso (groups, organizations), and micro (individuals). The SoS must be viewed as a multi-level enterprise system with four layered phenomena: social layer, institutional/economic layer, physical/process layer, and human layer [29]. Figure

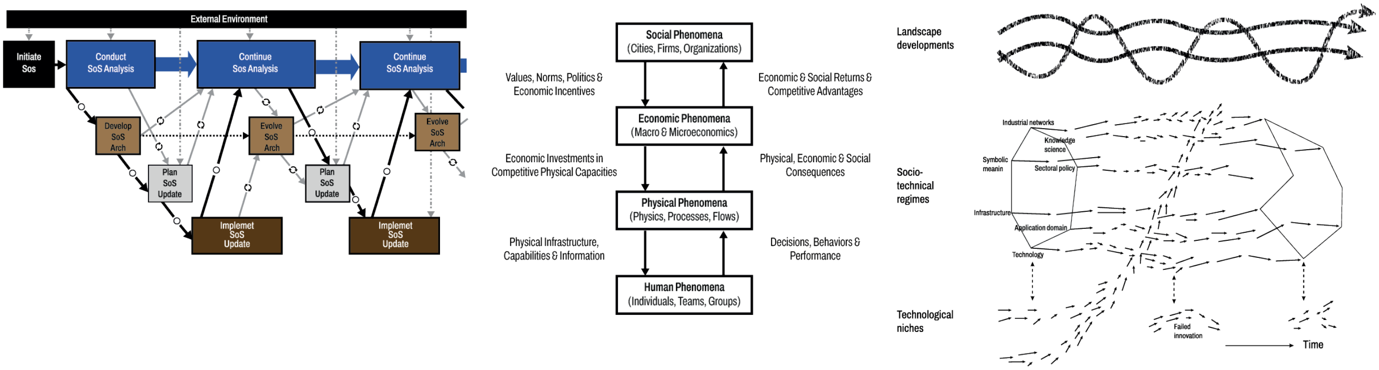

External Environment
Initiate Sos
Conduct SoS Analysis
Continue Sos Analysis
Continue SoS Analysis
Develop SoS Arch
Evolve SoS Arch
Evolve SoS Arch
Plan SoS Update
Plan SoS Update
Implemet SoS Update
Implemet SoS Update
Social Phenomena (Cities, Firms, Organizations)
Economic Phenomena (Macro & Microeconomics)
Physical Phenomena (Physics, Processes, Flows)
Human Phenomena (Individuals, Teams, Groups)
Values, Norms, Politics & Economic Incentives
Economic Investments in Competitive Physical Capacities
Physical Infrastructure, Capabilities & Information
Economic & Social Returns & Competitive Advantages
Physical, Economic & Social Consequences
Decisions, Behaviors & Performance
Landscape developments
Socio-technical regimes
Technological niches
Industrial networks
Knowledge science
Symbolic meanin
Sectoral policy
Infrastructure
Application domain
Technology
Failed innovation
Time

Figure 5. Multi-layer phenomena of SoS evolution.

5 shows the value of this type of analysis in both SoS Wave model planning and in transition management planning, which follow a similar layered set of changes. Implementation of SoS evolution is at the human and physical layers, while planning and execution are sociotechnical and socio-economic processes of the institutions that participate in the SoS. The multi-layer view directly supports both top-down and bottoms-up analysis of the phenomena that drive SoS evolution, particularly the human aspects of effective SoS accountability and responsibility.

With this in mind, we present two systems thinking tools that together aid in SoS analysis: the context analysis tool and the multi-level SoS evolution planning tool.

4.3.1. Context analysis tool

The context analysis tool in Table 4 [30] can be used to organize essential information reflecting desired SoS evolution outputs/outcomes, actors and institutions who will lead/oppose the evolution, activities they perform and how will they interact to affect evolution, and the enablers and barriers to the evolution process. These are each organized into the four multi-layer phenomena of Figure 5. The value of this tool is to institutionalize thinking about the multi-layer nature of SoS and the importance of analyzing enablers and barriers to change.

The additional value of the context analysis tool is its holistic collection of contributing factors to SoS evolution. It can serve as a running database of information useful in SoS evolutionary planning. It can capture important perspectives like “shared interests” (as enablers) and business relationships (as flows) in a form that is simple to use and maintain. To use this tool:

1. SoS outcomes represent value drivers to the stakeholders of the system derived from desired new SoS capabilities. Start with the outcomes column and list the outcomes of the SoS evolution at each level. This should focus on the new desired capabilities of the SoS and appropriate CS and their economic value at the “institutions” level. Above that in the table are any new outcomes expected of the domain. Below that in the table are any process level or human level outcomes that describe the evolution of the SoS.

2. Go back and capture the stakeholders who would support or oppose these outcomes as actors, and the enablers or barriers to evolution of the SoS. These would include accountable, responsible, consultant, or just informed stakeholders.

	Enablers and Barriers	Actors (and what they bring)	Interactions/ Activities	Outcomes
Social Domain	Governmental regulations, law, policy, infrastructure, environment, geography, economic conditions, security, global/national/regional investment strategies	Governments, nation-states, regions, infrastructure management organizations, cooperatives, non-government movements	Economic trends, demographic trends, shifts in governance, network attributes, ideologies, market trends	Buy-in and acceptance, employment shifts (macro), goal attainment, policy changes, new infrastructures, growth (macro), change (macro)
Institutions & Economics	Ownership, production environment/capacity, distribution networks, logistics capacity, import/export controls, standards	Government organizations, commercial organizations, universities, research institutions, lobby groups, standards organizations, unions, incubators	Market access & control, resource access, access to credit, technology transfer and patents, organizational relationships, ventures	Capacities, disruption and continuity, resilience, efficiency, change measures, strategies, product lines & systems, processes
Physical Phenomena, Processes & Flows	Legal structures, demand, pricing constraints, process constraints, standards, agreements, contracts, technology readiness/maturity	Leadership, work practices, organizational management structures, training, innovation practices, development practices, marketing practices, employment practices	Information flows, resource flows, knowledge transfer, financial flows, manufacturing flows, work activities, organizational interactions, customer interactions, market interactions	Products, buying, selling, technology uptake, market share/ leadership, innovations, collaborations, performance metrics
People & Assets	Culture, networks, community structures, access to information, proximity to resources and others, rights to property	People (individuals & classes), entrepreneurs, leaders, executives, skills, competencies, health, technologies, material resources, financial capital	Production, consumption/ purchasing, work automation, income, membership, teaming, learning	Use cases, tools & apps, employment, work roles, work outcomes, inclusion, health, standard of living, education, trust

Table 4. Context analysis tool [29].

3. Identify the interactions between stakeholders and "what they bring with them" (technologies, resources, etc.) and the activities that need to be accomplished to produce the desired SoS evolution. This column looks at processes, activities, and behaviors. Focus on statements that describe transformations that occur between entities in the system. For example, "making lower cost sensors" and "millions of miles of simulated driving" represent two phenomena helping to bring self-driving vehicles to the market. The first is a process and the second is a set of activities. Brainstorming and consultation with experts helps in this step.

It normally takes several stakeholder workshops to complete this tool. These are good opportunities to bring CS level stakeholders together to create shared interest around SoS evolution. Keep this table at a summary level, as it is conceived as a communications tool, and update it regularly and as needed to plan SoS evolution strategies.

The intent of this tool is to aid in planning all holistic aspects of SoS evolution, which must consider all aspects of the three-systems model and the sociotechnical and socio-economic multi-layer nature of large enterprises. Even if not actively using the tool, consider Table 4 as a useful guide to all of the types of phenomena that drive SoS evolution.

4.3.2. Multi-level SoS evolution planning tool

The multi-level SoS evolution planning tool in Table 5 is a means to visualize the details of transition management processes [31]. The tool integrates the primary decision information from all aspects of SoS analysis into a single framework that allows the team to visualize manageable transition paths in the broader SoS from lower-level CS organizations and systems. These paths are based on the desired changes sought within the problem space and the potential for innovation in the context of interest. The top half of the map outlines the SoS evolution context in terms of macro-level trends, events, and signals of change. The bottom half defines a SoS evolution in terms of possible innovations and relevant enablers and barriers of change, described at the CS level of the SoS.

As with the context analysis tool, using Table 5 starts with identifying the new desired SoS capabilities in the right-most column. This analysis is much less focused on the general context of the three-systems model and much more focused on the specific new desired capabilities (and priorities) of the SoS evolution.

The new desired capabilities (and priorities) of the SoS evolution are best described initially as a narrative that captures an "event" that deploys the new capabilities to the using community and the impact these changes will have to that capability. This is the "macro" view of the SoS outcomes. This event-based narrative is backed up by the SoS level and CS level trends that drive this narrative.

Each SoS evolutionary period starts with a current state, goes through an iteration period where the new capabilities "re-emerge" (there are often multiple iterations), and arrive at the desired end-state. It is these iterations that will provide the insight into SoS evolutionary pathways and help to organize the actual implementation of the new capabilities. These should be described first at the SoS (macro) level. There will be multiple iterations or "transitions" in SoS evolution, even though the table only represents one due to space constraints.

Starting at the bottom of the table, one then lists the individual CS-level change programs and actors in the current state, iterations, and desired end-state. Then moving up a row, the focus is on the institutions or organizations involved and the representative authorities and business models that would support SoS level change. A unique aspect of this table is the next two rows up, which capture enablers and barriers to SoS-level change. Because SoS leaders are accountable but may not have direct authority over CS updates, they typically can only enable them to happen or provide barriers to CS-level change that can have negative impact on the SoS-level performance.

4.4. Revisiting layers: Socio-economic focus of SoS evolution

When using these tools to conduct SoS analysis, it is important to apply a socio-economic frame of reference as opposed to a technical frame of reference. This implies one should focus on the Institutions and Economics related factors (and rows) in these tools. While SoS often exist in the Acknowledged and Collaborative categories, there must always be a lead organization that realizes economic value from the SoS capabilities and becomes the de facto change leader. As noted in Section 2.2, SoS develop and evolve in layers: technology, applications, information, and business.

One should also adopt a socio-economic layering of SoS [32]:

1. The set of organizations and independent individuals functioning as part of the SoS and how they are influenced. These live in the Institutions and Economics layer of the Context Analysis tool.
2. The infrastructure environment represented by various SoS-level technical foundations, institutions, regulations, procedures, and mechanisms. These live in the Social Domain layer of the tool.
3. The communication and logistics systems that provide the processes of interaction of the organizational components of the system. These live in the Physical Phenomena, Processes & Flows layer of the tool.
4. The innovation system: the set of activities, each of which is localized in space and in time, aimed at adapting the ecosystem to changes in the external environment. These live in the People and Assets layer of the tool.

What does this mean for SoS leadership and for leading implementation of SoS evolution? First, the implementation must start with the organizations involved and the economic value of the changes. From Table 5: "describe in detail the current SoS and CS level organizations as well as others in the context of use with descriptions of importance."

<table>
<tr><th></th><th>Current State</th><th>Signals of Change</th><th colspan="2">Iteration Period</th><th>Desired End State</th></tr>
<tr><th>SoS Level Events</th><td>List any past events that presented opportunities for SoS evolution.</td><td>What events signal to us that the SoS should change?</td><td colspan="2">Describe the plan for deployment of these changes and how they will be evaluated.</td><td>Provide a narrative of how these SoS capabilities were introduced and what impact they will have.</td></tr>
<tr><th>SoS Level Trends</th><td>What outcomes does the SoS provide and to whom? What CS level changes are currently in development or contemplated?</td><td>What are the current trends that signal new SoS capabilities are needed or becoming opportunities for evolution?</td><td colspan="2">Describe how you will integrate, test, deploy, and measure user adoption of the planned evolution.</td><td>What new outcomes should the SoS provide? What are the desired new capabilities? What CS need to be added, removed, or changed? How would you prioritize these capabilities?</td></tr>
<tr><th></th><th colspan="2">Current SoS primary capability</th><th colspan="2">Capabilities evaluated in the iteration</th><th>Desired SoS primary capability</th></tr>
<tr><th>Enablers</th><td>List enabling policy, technology, processes, organizations, etc. from the Context Analysis table.</td><td rowspan="2">What are the current trends that signal changes in an SoS enabler or barrier? New regulation, economic landscape, threat landscape, policy changes, etc.</td><td>Expected changes to enabling policy, technology, processes, organizations, etc. from the Context Analysis table.</td><td rowspan="2">Did the transition period indicate additional enablers of barriers would need to be addressed?</td><td>What changes to enabling policy, technology, processes, organizations, etc. are necessary to reach this desired state - from the Context Analysis table?</td></tr>
<tr><th>Barriers</th><td>List current potential barriers to evolution from the Context Analysis table.</td><td>Expected changes to potential barriers to evolution from the Context Analysis table.</td><td>What barriers need to be overcome to reach this desired state - from the Context Analysis table?</td></tr>
<tr><th>Institutional/ Organizational</th><td>Describe in detail the current SoS and CS level organizations as well as others in the context of use with descriptions of importance.</td><td rowspan="2">What are some new operational use concepts that the SoS could provide and who would be able to provide them?</td><td>What new SoS and CS level organizations will be involved? Who in the operational use concept will be involved?</td><td rowspan="2">How would you measure successful transition of the desired new SoS capabilities?</td><td>What are the new use cases for the desired capabilities and who will be able to provide them?</td></tr>
<tr><th>Example Programs and Involved Actors</th><td>Describe in detail the current SoS and CS level actors/ stakeholders and what you know about their work.</td><td>What CS level change programs or programs that produce new CS that you will integrate in this period?</td><td>What CS level change programs or programs that produce new CS would you plan to integrate and deploy in this end state?</td></tr>
</table>

Table 5. Multi-level SoS evolution planning tool [31].

Then move to the innovation system: "what CS level change programs or programs that produce new CS will you integrate in this period? Who produces them?" From there the leader must move to the leadership methods of Table 3: "what effort must be spent to establish and maintain actionable shared interest between CS teams and how will I communicate it across a large and possibly remote set of responsible teams?"

Third, evaluate or reevaluate the communications and logistics systems. This is where the enablers and barriers analysis of Tables 4 and 5 becomes most useful. As noted in Table 3: "a leader must create an environment where CS-level relationships can form and where responsible task owners can collaborate and become interdependent with respect to their task responsibilities."

The infrastructure environment is a primary developmental concern for new unprecedented SoS but is likely to be stable in evolution of a sustained SoS. Changes to the infrastructure environment should be discussed in the Trends and Events rows of Table 5 and where they are enablers or barriers to change, should be noted in those rows.

5. LEADERSHIP COMPETENCIES FOR SOS MANAGEMENT

When a new SoS is being created or an existing SoS is undergoing significant evolution, success will depend on the qualities of leadership. A framework for leadership in megaprojects and mega-systems that can apply well in the context of SoS evolution [33]. Four essential leadership capabilities are:

- Coordinating across diverse stakeholders toward shared outcomes.
- Addressing system complexity and uncertainty through learning.
- Creating flexibility in evolutionary strategies.
- Managing risk to SoS capabilities.

Management strategies tend to be more transactional and supported by transactional processes like "Plan-Do-Check-Act" (PDCA). In stability periods much of the SoS level change involves planning, implementing, checking results, and acting upon findings (the Wave model). In evolutionary periods, leadership can be described as more transformational. The Bass transformational leadership model defined two transactional and four transformational characteristics that contribute to high performance in leadership [34]. Figure 5 depicts how these six characteristics come together in a project setting.

While transformational leadership motivates and inspires followers, transactional leadership is more focused on "exchanges" between leader and follower in terms of work tasks, penalties, and rewards [35]. This framework is important to accountable SoS leadership because both transactional (project management) and transformational (influence and followership) leadership skills are necessary. The two transactional characteristics are more related to PDCA and include: management by exception (deal with misaligned CS level behaviors, plan and conduct SoS integrations, stabilize negotiated SoS relationships) and contingent reward (funds, incentives). The four transformational characteristics are more motivational and include: individualized consideration (coaching, delegation, training opportunities), idealized influence (model expected behaviors, use charismatic communication), inspirational commitment (gain commitment to the work and the SoS, teamwork), and intellectual stimulation (ensure the need for change,

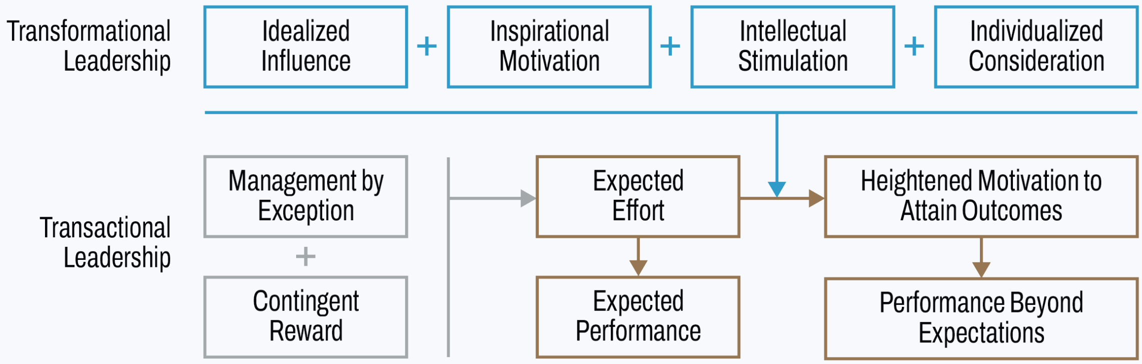


Figure 6. Bass transformational leadership model [33].

provide a strategy, build internal support, ensure external support, provide resources if needed, institutionalize changes) [35]. Transformational aspects of leadership are required to motivate responsible CS stakeholders to achieve SoS-level outcomes and to perform beyond their CS-level expected taskings.

In all the SoS case studies we reviewed, the creation and evolutionary periods of SoS had leaders with strong technical backgrounds in the SoS domain of interest, combined with experience sorting out difficult situations across complex supply chains. Critical skills for SoS leaders include [36]:

- Highly open to new experience, self-disciplined, engaging, stable, and test high in emotional intelligence.
- Project management remains important but in the context of cooperation and not in transactional methods (as it is usually employed).
- Preference for spending time on people management, alignment, and communications over work processes.
- Highly aware of their own abilities for learning.

Applying this model to the change practices of Table 3, we can characterize the leadership abilities of those that are accountable for SoS level outcomes:

Change Practice in SoS Evolution	Leadership skills [33]
There might not be a central authority for SoS evolution. There will be one or more leaders who are accountable for SoS outcomes and may have variable levels of control or influence over CS-level tasks that lead to CS-level evolution. The strategies to create and maintain SoS capabilities and outcomes will evolve with SoS evolution, and the accountable leaders may evolve with them. The direction of SoS evolution must be informed by a strategy.	• SoS leaders exhibit political savvy: the ability to exhibit confidence and professional diplomacy, while effectively relating to people at all levels internally and externally. • SoS leaders anticipate future situations, leaders with foresight can better plan for long-term outcomes and adapt to evolving project environments. • SoS leaders embrace uncertainty and ambiguity. • They are self-aware and self-motivated: they continually reassess their own judgment and decisions, while showing enthusiasm, passion, and a strong commitment to SoS goals. • They are open-minded: they are willing to revise their plans based on new information in order to achieve better SoS outcomes. • They are courageous and willing to make tough decisions, address underperformance, and challenge unrealistic expectations or assumptions.

Change Practice in SoS Evolution	Leadership skills [33]
SoS must be built on shared interests and/or concerns. Efforts from a stakeholder that has a leadership position must be spent to establish and maintain shared interest. Leaders must find actionable shared interest between CS teams and communicate it across a large and possibly remote set of responsible teams.	• Leaders with strong technical and domain skills in the SoS are needed for creating shared interest. • SoS leaders must be willing to learn along with CS responsible stakeholders. • SoS leaders must be trustworthy and establish strong leader-follower relationships along lines or shared interests. • Strategic thinking is required for setting realistic goals, and ensuring alignment between SoS and CS objectives. • Vision/Goal Setting: a clear vision is crucial for guiding decision-making and maintaining stakeholder alignment. • SoS leaders master all aspects of effective communication and make it a focus of their work. Communication must be at a higher, more visionary level in SoS leadership.
A person that has a leadership position with respect to the SoS, who can be either responsible or accountable, must create an environment where CS-level relationships can form and where responsible task owners can collaborate and become interdependent with respect to their task responsibilities. This interdependence serves to distribute power across CS and must be an active process when SoS capabilities are being updated or evolved. This includes engaging with consultants on behalf of the CS and keeping other stakeholders informed.	• SoS leaders encourage a culture of openness and communication and are themselves open to new ideas and perspectives. • SoS leaders build cultures of collaboration across CS-level teams. • SoS leaders create an environment that actively shows they believe responsible task owners act in good faith and do their best to help the SoS achieve its goals. • SoS leaders coach and mentor others independent of CS organization; they help to develop others' leadership skills, foster collaboration, and enhance trust and performance. • SoS leaders build relationships; they create and maintain connections with people to create a sense of team.
Shared interests are necessary but insufficient to succeed at SoS work tasks when CS level priorities interfere with SoS work responsibilities. Incentives for SoS level work must be applied and captured into documented agreements that are negotiated between CS, whether formal contracts or other more indirect business relationships. These incentives may be economic, partnership-based, defined in policy, etc.	• There will be conflicts between stakeholders and conflicts of priority within the team. Managing stakeholders calls for negotiating and diplomatic skills. SoS leaders need to be able to bring stakeholders together on critical SoS level decisions. • SoS are complex, involve a wide variety of stakeholders, and require the integration of multiple disciplines. SoS leaders help solve complex problems. They understand knowledge exchange and transfer within and across social networks is critical and they actively visualize knowledge to develop shared understanding across stakeholders. • SoS leaders create an environment that encourages and prioritizes gaining new knowledge and skills. Learning is rewarded. • They are highly experienced in managing complex supply chains across multiple CS. • They are extremely business savvy. They know when to become more directed, using explicit contracts with CS for SoS requirements. Yet they create flexibility in contracts and business relationships that incentivize integration problem solving over simple cost and schedule factors. • They harness network effects, effectively using policy, standards, or common economic interest [22].

6. CONCLUSION

SoS evolution is a leadership challenge that spans two types of SoS phenomena: the emergence of a new SoS and the sustainment of an existing SoS. One should view this as a sociotechnical and socio-economic process, not just new capabilities arising from an engineering process. Analysis and implementation of SoS evolution must start with the organizations involved and the economic value of the changes. From there evolution will arise from the innovation system, which is separate from the SoS of Interest, and the leaders' abilities to create shared interest across multiple (primarily engineering) teams. All SoS must be led, although some of that may be informal. The leader must create an environment where CS-level relationships can form and where responsible task owners can collaborate and become interdependent with respect to their task responsibilities. Finally, SoS leaders must be holistic and able to place emerging trends and events that will affect SoS capabilities or business models into a transition plan. There are a unique set of leadership skills associated with SoS, and individuals with these skills tend to arise in development of new SoS and should be trusted and nurtured.

REFERENCES

1. M. Maier, "Architecting Principles for Systems of Systems," Proc. of the Sixth Annual International Symposium, International Council on Systems Engineering, Boston, MA, (1996), pp. 567- 574.

2. International Standards Organization (ISO)/ Institute of Electrical and Electronics Engineers (IEEE), ISO/IEC/IEEE 21840:2019(E) Systems and software engineering - Guidelines for the utilization of ISO/IEC/IEEE 15288 in the context of system of systems, 2019.

3. International Standards Organization (ISO)/ Institute of Electrical and Electronics Engineers (IEEE), ISO/IEC/IEEE 21839:2019(E) Systems and software engineering - System of systems (SoS) considerations in life cycle stages of a system, 2019.

4. M. Jamshidi, "System-of-Systems Engineering – a Definition," IEEE SMC (October 2005).

5. J. Dahmann. 2014. Systems of systems pain points. INCOSE International Symposium on Systems Engineering, 2014.

6. C. B. Nielsen, P. G. Larsen, J. Fitzgerald, J. Woodcock, and J. Peleska, Systems of Systems Engineering: Basic Concepts, Model-Based Techniques, and Research Directions. ACM Comput. Surv. 48, (2), September 2015.

7. J.H. Holland (2006). "Studying Complex Adaptive Systems". Journal of Systems Science and Complexity. 19 (1): 1–8.

8. J. Boardman and B. Sauser, "System of systems - The meaning of 'of,'" In 2006 IEEE/SMC International Conference on System of Systems Engineering. IEEE.

9. McDermott, T. and Nadolski, M. 2017. "Emergence as Innovation in Systems of Systems - a Three Systems Model," 12th International Conference on System of Systems Engineering (SoSE), Waikoloa, US-HI.

10. R. Abbott, "Open at the top; open at the bottom; and continually (but slowly) evolving," In 2006 IEEE/SMC International Conference on System of Systems Engineering. IEEE.

11. Y. Bar-Yam, M. A. Allison, R. Batdorf, H. Chen, H. Generazio, H. Singh, and S. Tucker. 2004. The characteristics and emerging behaviors of system of systems. NECSI: Complex Physical, Biological and Social Systems Project, 1–16.

12. Rogers, Everett. Diffusion of Innovations, 5th Edition. Simon and Schuster. 16 August 2003.

13. R.M. Henderson, K.B. Clark, "Architectural Innovation: The Reconfiguration of Existing Product Technologies and the Failure of Established Firms," Administrative Science Quarterly, Vol. 35, 1990.

14. J. Dahmann, G. Rebovich, J. Lane, R. Lowry and K. Baldwin, "An implementers' view of systems engineering for systems of systems," 2011 IEEE International Systems Conference, Montreal, 2011, pp. 212-217.

15. F.W. Geels, "Technological transitions as evolutionary reconfiguration processes: A multi-level perspective and a case-study," Research Policy, December 2002, 31 (8-9), 1257-1274.

16. J. Rotmans, R. Kemp, & M. van Asselt. “More evolution than revolution: transition management in public policy,” Foresight 3 (1).

17. A. Mostafavi, D. M. Abraham, D. DeLaurentis, & J. Sinfield, “Exploring the Dimensions of Systems of Innovation Analysis: A System of Systems Framework,” IEEE Systems Journal, 2011, 5 (2), pp. 256-265.

18. State of California, Senate Bill No. 1298, Chapter 570, Approved by Governor on Sepember 25, 2012. Retrieved 4/2025 from http://www.leginfo.ca.gov/pub/11-12/bill/sen/sb_1251-1300/sb_1298_bill_20120925_chaptered.pdf.

19. H. Lawson, A Journey Through the Systems Landscape. London: College Publications, 2010.

20. Gary T. Anderson Opportunities for Innovation in a System of Systems Framework, Proceedings of the 2009 IEEE International Conference on Systems, Man, and Cybernetics San Antonio, TX, USA - October 2009.Sage, Andrew P. and Cuppan, Christopher D. ‘On the Systems Engineering and Management of Systems of Systems and Federations of Systems ’. 1 Jan. 2001 : 325 – 345. Print.

21. Sage, Andrew P. and Cuppan, Christopher D. ‘On the Systems Engineering and Management of Systems of Systems and Federations of Systems ’. 1 Jan. 2001 : 325 – 345. Print.

22. Waymo. Wikipedia. Retrieved 4/2025 from https://en.wikipedia.org/wiki/Waymo.

23. Mohamed Hichem Fendali, Isabelle Borne, Djamel Meslati, Labiba Souici-Meslati. “How to steer evolution of a system-of-systems” Systems Engineering. 2024;1–17. ©2024Wiley Periodicals LLC.

24. Project Management Institute (PMI). A Guide to the Project Management Body of Knowledge (PMBOK Guide) (2004 edition). Newtown Square, PA: Project Management Institute.

25. Smith, Michael L.; Erwin, James. “Role & Responsibility Charting (RACI)” (PDF). Project Management Institute California Inland Empire Chapter. p. 5. Archived from the original on 16 September 2022. Retrieved 18 May 2023.

26. Rouse, William B (Ed.). Enterprise Transformation: Understanding and Enabling Fundamental Change. Wiley 2006.

27. J. Lane, Planning for SoS Evolution, presentation to the San Diego Chapter of INCOSE 23 July 2016. Retrieved 4/2025 from https://sdincose.org/wp-content/uploads/2016/10/SD-INCOSE-Tutorial-Planning-for-SoS-Evolution.pdf

28. W. B. Rouse, “Policy Flight Simulators For Transforming Large-Scale Public-Private Enterprises;” Third International Engineering Systems Symposium CESUN 2012; 18-20 June 2012.

29. W. B. Rouse & D. Bodner, Multi-level modeling of complex socio-technical systems – phase 1, A013 - final technical report, SERC-2013-TR-020-2, Systems Engineering Research Center.

30. McDermott, Tom. “Applying Systems-of-Systems Principles to Purposeful Design of Human Systems.” 2018 13th Annual Conference on System of Systems Engineering (SoSE) (2018): 150-156.

31. Kemp, Loorbach, & Rotmans; “Transition Management as a model for managing processes of co-evolution towards sustainable development;” International Journal of Sustainable Development & World Ecology 14(2007) 1-15.

32. Kleiner, G. “Socio-economic ecosystems in the light of system paradigm.” System analysis in economics-2018, pp. 4-12. 2018.

33. T. McDermott & N. Hutchison, Critical skills for successful leadership of large complex projects, preprint, 22nd Annual Acquisition Research Symposium & Innovation Summit, 2025.

34. Aarons, G.A. Transformational and transactional leadership: association with attitudes toward evidence-based practice. Psychiatr Serv. Aug 2006; 57(8):1162-9.

35. Van Wart, M. Evaluating Transformational Leaders: The Challenging Case of Eric Shinseki and the U.S. Department of Veterans Affairs. Public Administration Review, 75(5): 760-769. September|October 2015.

36. Merrow, E., & Nandurdikar, N. Leading Complex Projects. John Wiley & Sons. 2018.

BIOGRAPHIES

TOM MCDERMOTT

Tom McDermott is the Chief Technology Officer of the Systems Engineering Research Center (SERC) and a faculty member in the School of Systems and Enterprises at Stevens Institute of Technology in Hoboken, NJ. With the SERC he develops new research strategies and is leading research on digital transformation, education, security, and artificial intelligence applications. He previously held roles as Faculty and Director of Research at Georgia Tech Research Institute and as Director and Integrated Product Team Manager for the F-22 Raptor Avionics team at Lockheed Martin. Mr. McDermott teaches system architecture, systems and critical thinking, and leadership. He provides executive level consulting as a systems engineering and organizational strategy expert. He is a fellow of the International Council on Systems Engineering (INCOSE) and recently completed 3 years as INCOSE Director of Strategic Integration.

MIGUEL ÁNGEL COLL MATAMALAS

Miguel Ángel Coll Matamalas is a Systems Engineer at Isdefe, currently supporting the Logistics Support Command of the Navy in the development of logistic doctrine and activities related to obtaining Integrated Logistic Support for ongoing Acquisition Programs. In particular, he is involved in the development of the Digital Twin of the F-110 frigates. He also participates in the NATO Working Group responsible for reviewing the ALP-10 standard (NATO Guidance for Integrated Life Cycle Support). Miguel Ángel is a Naval and Oceanic Engineer from the Universidad Politécnica de Madrid and holds an Executive MBA from IESE (University of Navarra). In his previous career, he has occupied several roles associated with maintenance and asset management, spanning both the maritime industry and water infrastructure sectors. With the aim of applying this experience in different areas of Sustainment, Miguel Ángel has a special interest in improving the supportability of systems through influence in design; in the development of digital twins, prescriptive maintenance, and Model-Based Product Support (MBPS).

CHAPTER 9

Novel analysis, modeling, and simulation methods for SoS

José Luis de Rosario, *Isdefe (jlderosario@isdefe.es)*
Dr. Paul Grogan, *Arizona State University (paul.grogan@asu.edu)*
Dr. Alessandro Golkar, *Technical University of Munich (golkar@tum.de)*
Dr. Amro Farid, *Stevens Institute of Technology (amro.farid@stevens.edu)*
Dr. Young-Jun Son, *Purdue University (yjson@purdue.edu)*
Dr. Nil Egin, *The Pennsylvania State University (nhe2@psu.edu)*
Dr. John Little, *Virginia Tech (jcl@vt.edu)*

Abstract

This chapter presents a survey of novel analysis and simulation methods that are useful to engineer SoS, including: coordination of systems, federation of systems, heterofunctional modeling, federated modeling and simulation, agent-based modeling, and SoS approaches to modeling.

Keywords

Coordination of systems, federation of systems, heterofunctional modeling, federated modeling and simulation, agent-based modeling.

1. INTRODUCTION

As discussed throughout the book, the engineering of Systems of Systems (SoS) presents a distinct set of challenges that transcend those faced in traditional systems engineering. SoS are characterized by the operational and managerial independence of their constituent systems, evolutionary development, emergent behavior, and often, heterogeneity in function, purpose, and design. These characteristics demand analysis and simulation methods that can accommodate dynamic interactions, decentralized control, and multiple layers of abstraction.

This chapter surveys a collection of novel analysis and simulation methods that have emerged or matured in response to the specific demands of SoS engineering. These methods provide engineers with tools to represent, reason about, and test SoS concepts in ways that traditional approaches cannot support effectively. The chapter covers approaches that facilitate the coordination of systems to achieve shared objectives, and those that support the federation of systems while preserving local autonomy. It explores heterofunctional modeling techniques that account for the diversity of roles and interactions among systems, and presents federated modeling and simulation strategies that allow independent models to interoperate coherently. The chapter also covers agent-based modeling, a powerful method for capturing the behavior of autonomous entities and their interactions, and broader SoS-specific modeling approaches that provide tailored abstractions and formalisms to capture the complexity of these large-scale engineered constructs.

By surveying these methods, this chapter aims to provide SoS engineers with a roadmap to select, combine, and apply suitable analysis and simulation techniques throughout the lifecycle of a SoS.

2. COORDINATION OF SYSTEMS

2.1. Introduction

Systems engineering is traditionally a centrally managed approach to the successful design, integration, testing, operation, and retirement of engineered systems. Centralized approaches help to address system-level objectives by directing engineering efforts to consider emergent properties that may not fall within the scope of individual subsystem teams. Characteristic of this point, recent interest in model-based systems engineering (MBSE) approaches emphasize centralized information systems to collect, manage, and share design knowledge across a design team [1].

SoS challenge conventional concepts of control in systems engineering because no single system-level actor possesses complete authority over others. SoS engineering problems exhibit more decentralized structures, which influence the type and nature of supporting methods. The seminal work by Maier describes architecting principles for SoS, referred more precisely as collaborative systems, that emphasize more indirect control via stable intermediate forms, policy triage, leveraging at the interfaces, and incentives to ensure cooperation [2].

The type and degree of interaction within a SoS spans a continuum of collective endeavors [3]. Coordination is a loosely coupled information sharing and planning paradigm that pursues individual objectives by reducing conflict. Cooperation is a moderately coupled resource sharing paradigm that pursues mutual objectives with shared outcomes. Finally, collaboration is a tightly coupled united paradigm that pursues a joint objective that individuals are incapable of achieving alone. Advances in SoS modeling and analysis techniques seek to promote design actions leading to mutually desirable collective outcomes while mitigating undesirable ones.

2.2. Modeling coordination with game theory

Game theory is an economic field that studies strategic interaction. It builds upon key models and methods of decision theory, which itself studies individual decision-making under uncertainty. The key distinction between game theory and decision theory hinges on the consideration of interactive decision-making processes among actors who iteratively consider each other's perspectives about the coupled problem when devising decision-making strategies. Game theory is particularly applicable to SoS problems because it explicitly considers decentralized control over actions and multiple, potentially completing, actor objectives.

Fundamental game theory constructs define the actors, their available actions, and resulting shared outcomes. Each actor possesses a utility-based measure of preference for outcomes. Preferences follow a similar logic to objective functions in value-driven design [4]; however, coupled outcomes evaluated by each actor-specific utility function can lead to conflicting preferences for outcomes.

An actor's *strategy* describes the complete process by which they select an action. Pure strategies select one action with certainty (thus, actions are also commonly referred to as strategies) and mixed strategies select probabilistic combinations of actions. To identify rational strategies (i.e., those leading to preferred outcomes), game theory explicitly considers strategic interaction among actors, rather than simply treating other's actions as sources of exogeneous uncertainty.

There exist a variety of game theoretical problem formulations and analysis methods; however, this section presents a basic one: a normal form game. It is represented by a matrix that quantifies preference for outcomes (payoffs) for each actor resulting from simultaneous action combinations, often over discrete action spaces. Normal form games typically model single-stage decision problems; however, repeated games can also study strategy of sequential decisions. Non-cooperative games focus on the baseline decision-making dynamics arising from payoff values while cooperative games allow for within-game agreements to modify such values.

Game theory analysis methods focus on identifying *equilibrium* strategies. A Nash equilibrium is a strategy set where no actor has individual incentive to deviate to a different strategy. It represents a potential "solution" to a game because it finds a naturally stable decision point that is immune to interactive effects. Games may have zero, one, or many equilibriums and actors may have different preferences for each equilibrium. The presence of an equilibrium does not suggest it is mutually desirable to all actors.

Rather than solving realistic day-to-day problems, game theory strives to anticipate strategic behavior by understanding the underlying forces that shape actions in more abstract settings. Many game theory problems consider a small set of actors and their actions. A surprisingly rich variety of strategic dynamics emerge from games with two actors and two actions based on the number and nature of equilibrium strategies present. For example, the famous prisoner's dilemma game is a two-actor, cooperate-or-defect game that exhibits one equilibrium strategy (both defect), which does not lead to the most desirable outcome (both cooperate). The tension between individual and collective benefits helps to understand the strategic dynamics behind many collective endeavors.

In addition to identifying equilibrium strategies, game theory methods also provide comparative analysis of strategies within or across related games. For example, a stag hunt is a two-actor, collaborate-or-defect game that exhibits a bistability dynamic where the two equilibrium strategies (both collaborate or both defect) trade sources of risk and reward. Game theory analysis of risk dominance evaluates comparative attractiveness of the two strategies by measuring the relative stability of collaboration, which helps to identify and pursue robust collaboration opportunities [5].

Broader applications of mechanism design leverage game theory analysis to propose structural game changes that are anticipated to result in more collectively desirable outcomes. In this setting, cooperative game theory allows enforceable agreements within actor coalitions to shift focus away from maximizing individual payoffs to sharing collective payoffs. As in most design problems, mechanism design operates within a vast space of creative design solutions that balance competing factors such as economic efficiency and equity.

2.3. Illustrative example: satellite coordination

Consider an example SoS coordination problem with two independent space agencies: Red and Blue. Red operates a small research satellite and Blue operates a large meteorological satellite. As an example of bilateral coordination, the agencies must decide on actions to mitigate the threat of a collision in response to a hazardous conjunction event.

Both actors have two available actions: do nothing or maneuver their satellite, leading to four possible outcomes. Table 1 illustrates the normal form representation of this coordination game using notional utility (payoff) values. Note that the mixed strategy has been added as an additional row and column to aid discussion. Utilities quantify the expected utility for each outcome which itself may compose a distribution of events (e.g., the outcome where both actors do nothing may only probabilistically lead to a collision). Financial units (e.g., USD $M) provide a relatively unbiased proxy for utility despite being affected by variable human risk attitudes. For simplicity, this example assumes risk-neutral decision-making. The outcome where both actors do nothing results in severe negative consequences denoted as -$50M for Red and -$100M for Blue. The other outcomes result in slightly negative consequences due to propellant consumption that reduces satellite operational life, shown as -$2M for Red and -$3M for Blue. Utility values are not generally comparable between actors due to differences in preferences; however, the selected quantities are intended to highlight asymmetries between the two actors.

		Blue Agency		4% Do Nothing 96% Maneuver
		Do Nothing	Maneuver	
Red Agency	Do Nothing	-50 -100	0 -3	-2 -6.88
	Maneuver	-2 0	-2 -3	-2 -2.88
3% Do Nothing, 97% Maneuver		-3.44 -3	-1.94 -3	-2 -3

Table 1. Satellite collision coordination problem (Utilities in USD $M).

Equilibrium analysis searches for stable strategy sets, revealing two pure equilibria and one mixed equilibrium. The first pure equilibrium strategy occurs if Red chooses to do nothing, and Blue chooses to maneuver. Under this baseline, Red would not unilaterally choose to maneuver, because the threat is already mitigated. Similarly, Blue would not unilaterally choose to do nothing, because it results in a worse outcome than proceeding with a maneuver. The second pure equilibrium strategy occurs for the similar scenario where Red chooses to maneuver, and Blue chooses to do nothing with the same justification. The mixed equilibrium produces indifference to strategy selection. In this game, when Red chooses to maneuver with probability (-100+3)/(-100-0+3-3) = 0.97, Blue has expected utility -$2M for all outcomes. When Blue chooses to maneuver with probability (-50-0)/(-50+2-0-2) = 0.96, Red has expected utility -$3M for all outcomes.

Actors do not have equal preferences for the three equilibrium solutions. Red prefers the equilibrium where Blue always maneuvers. Similarly, Blue prefers the equilibrium where Red always maneuvers. The mixed equilibrium is not preferred by either actor. Game theory provides little more progress towards a solution and external factors such as negotiation or brinksmanship may be required to resolve the strategic tension. Indeed, this example is an instance of games variously described as chicken, hawk-dove, or snowdrift that represent a theoretical model of intrinsic conflict.

Cooperative game theory permits enforcement of agreements that materially impact outcomes. In the above problem, it is comparatively less costly for Red to maneuver their small satellite than Blue; however, Red bears the cost of maneuvering their own satellite. Mechanism design proposes incentives to promote efficient solutions. For example, consider a mechanism where Blue agrees to compensate Red for maneuvering their smaller satellite to mitigate a potential collision, regardless of Blue's action. Table 2 represents the value exchange as a deduction of Y from Blue and an addition of X to Red. Note that X and Y may not be equal if Red and Blue differently value the resource being exchanged or if transactional friction erodes a part of the value.

		Blue Agency	
		Do Nothing	Maneuver
Red Agency	Do Nothing	-50 -100	0 -3
	Maneuver	X-2 0-Y	X-2 -3-Y

Table 2. Satellite collision coordination problem with incentives.

For values X > $2M, only one equilibrium strategy remains where Red always chooses to maneuver and Blue always chooses to do nothing. Compared to the original game, this solution is better than Red's best-case equilibrium and better than Blue's worst-case equilibrium for values Y < $3M. Additionally, this solution avoids the strategic uncertainty associated with multiple equilibriums. However, a new potential problem arises: Red now has a perverse incentive to benefit from collision hazards as a source of additional revenue. An expanded system scope and additional analysis will be required to devise coordinated SoS solutions that resist exploitation.

3. FEDERATION OF SYSTEMS

3.1. Introduction

Federations of Systems (FoS) are a class of SoS that may operate during their life cycle independently or may voluntarily cooperate with each other with the intent of obtaining mutual benefits. Unlike hierarchical or tightly integrated systems, federations are a class of distributed systems that maintain the management, operational, and goal independence of their constituent systems. At the same time, they are designed in such a way as to promote collaboration among them. The FoS paradigm is particularly relevant in scenarios where resource sharing can improve value delivery while addressing cost and complexity challenges-particularly for systems characterized by high capital costs and/or high recurring operating costs.

The characteristics defining FoS are outlined in Table 3. The motivation behind FoSs lies in the potential for improved resource utilization, and the possibility of providing added value by promoting collaboration among traditionally isolated systems. Collaboration between systems is particularly important in exploiting opportunities arising from the underutilization of resources.

SoS employ mechanisms structured to ensure integration when required. Such integration is implemented through centralized communication protocols. Governance frameworks are also possible to enable flexible collaborations, often of an opportunistic nature.

One instance of system federations is represented by the Federated Satellite Systems (FSS) paradigm [7]. Satellites in FSS cooperate to exploit resources that would otherwise go unused because they are not needed at particular points in their operational life. Examples of resources that can be shared in a federation include bandwidth, data storage, as well as computational capacity. Participation in a federation makes it possible to reduce operational inefficiencies and thus make better use of the capacity in orbit –an equally important goal for sustainable and responsible space. One of the problems that arise in implementing the idea of federations in orbit is that of the mechanisms necessary for their operations. In other words, it is necessary to define operational arrangements and incentives of an economic or utilitarian nature that enable satellites to operate efficiently while maintaining their autonomy. One of the mechanisms considered for this purpose is that of constrained– bid reverse auctions [8]. This approach enables the creation of a commercial ecosystem of in-orbit space assets, therefore generating opportunities for innovation and improvement in the capabilities available to individual satellites.

The study of system federations is interdisciplinary in nature and is not limited to the space sector alone. We can easily postulate other examples of possible federations in different industrial fields. For example, in the energy sector, it is possible to conceive of federations of energy systems such as wind turbines and solar farms, as well as battery storage units. In this case, electricity and its storage are the resources made available. The objective of the federation thus becomes to make excess energy available, promoting the balancing of the grid as well as the pricing dynamics between supply and demand. Further impacts of such an approach include the possible reduction of operating costs and minimization of

Defining the characteristics of system federations			
Voluntary cooperation: Participation based on mutual agreement, incentivized by the individual goals of the systems involved.	**Mutual benefits:** federations motivated by the emergence of mutual benefits from collaboration. Examples of mutual benefits are improved performance, reduced costs, or enhanced capacity.	**Independent management:** Limiting collaboration between systems to areas of mutual interest, leaving each system with its own autonomy.	**Alignment of operational goals:** maximizing the federation's group goals, subject to operational independence and the pursuit of individual goals.

Table3 Characteristics of Federations of Systems (FoS).

environmental impact through efficient utilization in terms of energy storage, transportation, and distribution. An additional area of interest for federations is the telecommunications sector. In the latter, federated wireless networks can enable independent service providers to share their resources, such as base stations and bandwidth, in order, for example, to expand coverage and improve service reliability.

The motivation behind the idea of system federations is the need for resource efficiencies in the face of constraints of technical complexity and economics that are often difficult for complex systems such as those in high capital cost industries. Through the promotion of collaboration among independent entities, system federations allow the capabilities of individual systems to be increased at a lower marginal cost than would be possible in the absence of collaborative opportunities. This approach also enables the reduction of operational costs and the balancing of resource utilization when designed within a sharing network. In this chapter, we will explore the basic principles, challenges, and practical applications of system federations. We will use federated satellite systems as a case study of FoS. The goal of the chapter is to provide a comprehensive discussion on the potential for system federations have to be a viable approach to optimizing the efficiency of systems operating in networks, as well as an approach to promoting scalability and sustainability in resource utilization.

3.2. Fundamental concepts of federations

The concept of *synergy* is central to the value proposition of system federations. In this context, synergy is defined as the aggregate benefit resulting from the cooperation of systems operating within a federation. Synergy is defined as the mutual benefit resulting from the cooperation of independent systems. Such cooperation enables the participating systems to collectively achieve greater results than could be achieved in isolation [9]. Mathematically, synergy is defined as follows:

$$\begin{aligned} S_{y_{N\leftrightarrow N}} &= \Sigma_i^N \Delta C_i^j \\ \Delta C_i^j &= F_i\left(U^j\right) - C^j \end{aligned} \qquad (1)$$

Where $S_{y_{N\leftrightarrow N}}$ is the synergy between a set of *N* systems, andΔC_i^j is the cost delta of a specific architecture *j* for a given system *i*. ΔC_i^j quantifies as the difference between the Pareto frontier of the system trade space represented by the cost-utility mapping $F_i\left(U^j\right)$, and the cost of the architecture C^j . Positive synergy occurs when the sum of the delta cost functions is positive, meaning that the net improvement in cost-benefit performance has been achieved across the federation and is treated as a heuristic to guide architectural decisions. Synergy emerges from correlated changes in the design spaces of federated systems, as seen in their utility-cost tradeoffs [4]. In federated satellite systems (FSS), for example, sharing bandwidth or relay capabilities improves the performance of Earth observation missions while reducing costs. Improvements in the local utility of one system directly affect the cost-benefit performance of others, forming a network of interdependencies [10].

Tradespace exploration is an established approach used for performance analysis of system federations. This methodological approach evaluates the space of possible solutions in order to identify system architectures with optimal tradeoffs between cost, performance, and other decision variables of interest [11, 12]. The exploration of tradespace allows systems that are part of (or candidates to be part of) a federation to explore the value added by their participation in the pooled resource collective. For example, in FSS, tradespace analysis is used to assess the cost-benefit sharing of bandwidth and data processing capabilities. This purely analytical approach allows for a quantitative assessment of the conditions under which federations are most beneficial and identifies scenarios in which they may not provide the expected benefits.

Not in all cases can system federations be considered beneficial. Indeed, they also present unique challenges at the system architecture level. One of the main challenges is interoperability among the independent systems that make up the federation [13]. Interoperability requires algorithms, hardware, and software, the complexity of which may outweigh the possible benefits of collaboration. Examples of elements needed for interoperability include communication protocols, as well as standardization of data formats and interfaces. Standardization itself is an expensive and complex process of coordination among heterogeneous entities. In addition, the decentralized nature of FoS complicates the creation of universal standards, as individual systems may prioritize local optimization over global compatibility.

Another architectural challenge is to assess the scalability potential of federated systems. The number of possible interactions and dependencies among federated systems increases exponentially as the number of systems participating in the federation increases. For this reason, offering performance or reliability guarantees becomes particularly challenging, particularly in the face of the operational uncertainties that such systems face. A critical consideration in this regard is the balance between the costs of creating and maintaining interfaces and the expected benefits of cooperation. The management of interfaces that enable the interaction of federated systems is a key issue that has been addressed in the context of federated satellite systems [9]. The two main strategies considered are direct changes, such as replacing or adding interfaces, and indirect changes, involving middleware or negotiator nodes [14]

Direct interface modifications often require physical access to satellites, which is highly impractical for spacecraft not specifically designed for in-orbit maintenance [15]. The latter issue has raised particular interest in recent years, especially in the context of modular approaches to space system design [16]. The challenge of in-orbit maintenance of satellites is not new and has been addressed several times during the evolution of the space industry. One example is that Hubble Space Telescope, which was designed with modular components to enable regular maintenance missions. The telescope's design allowed astronauts to repair defects, such as the initial spherical aberration of the primary mirror, and to upgrade instruments and subsystems, significantly extending its operational lifetime and scientific utility [17]. However, traditionally on-orbit maintenance has entailed significant costs that are justifiable only for flagship space missions of Hubble's caliber, and thus not affordable for the majority of space missions. This may change in the near future thanks to reduced launch costs and advances in in-orbit servicing technology [18]. An additional challenge to satellite interfacing is the regulatory constraints and heterogeneity of communication protocols between satellites. Careful evaluation of these challenges is needed to determine whether the costs and risks of retrofitting existing satellites outweigh the expected benefits.

Indirect modifications, such as the use of *negotiator nodes,* offer a more feasible approach to enabling interoperability [14]. Negotiators act as intermediaries that facilitate resource sharing and communication between satellites, such as through the use of Software Defined Radio (SDR) technologies. The work presented in [14] evaluates the concept of hosted payloads or independent negotiator satellites equipped with reconfigurable antennas and SDR. These digital technologies allow dynamic adaptation to different communication protocols and frequencies, thus addressing the heterogeneity of systems in a federation. However, even with this approach, the trade-off between the added complexity introduced by negotiators and the benefits of increased collaboration must be carefully analyzed.

The case of federated satellite systems shows that, given the right operational and economic conditions, interface integration can unlock significant synergies, enabling systems to achieve collective goals that exceed their individual capabilities [7].

3.3. Challenges and opportunities in system federations

Federated engineering systems face challenges arising from their decentralized nature and the inherent independence of constituent systems. These challenges include trust and security. These are particularly critical as they affect the willingness of stakeholders to participate in and maintain federation. Trust issues in FoS stem from problems with data authentication, integrity, and confidentiality. Without robust mechanisms to address these issues, the risk of data tampering, unauthorized access, and misinformation increases, potentially undermining the collaboration necessary for a successful federation.

One proposed solution to these challenges is the adoption of blockchain systems [19]. Blockchain ledgers together with data encryption are a potential solution to manage exchange of data and metadata, through transparent mechanisms that are able to ensure data privacy when necessary. Another useful feature of blockchain in this context is its tamper-proof nature that allows for providing proofs on data sources and transactions. As a result, stakeholders have trust guarantees and in the context of FSS, blockchain can be used to securely record resource sharing transactions, such as bandwidth or data processing exchanges, ensuring that all parties have a verifiable record of agreements and deliveries.

A complementary approach to blockcahin is represented by of Public Key Infrastructure (PKI) protocols. PKI enables secure communication by leveraging cryptographic keys for authentication and encryption. In the context of federations, this approach allows for establishing a secure trust basis by allowing stakeholders to verify the identity of communicating parties and ensure the integrity of shared data. For example, in federated satellite systems, PKI can facilitate secure inter-satellite communication by ensuring that only authenticated parties participate in resource sharing operations and that exchanged data remains confidential and untampered with [20].

In addition to trust and security, federated systems face the challenge of dynamic network topologies and uncertain demand and supply for resources since nodes can give and revoke their sharing availability on an individual basis. Membership in a federation can be uncertain, and therefore volatile. This volatility poses challenges for coordinating, balancing, and optimizing operations. One potential solution to this issue is represented by Markov decision processes (MDPs). MDPs are a tool for modeling system operations under conditions of uncertainty [21] and can be used for predicting and optimizing federation behavior under uncertainty. Operators can analyze the possible states of the system using MDPs, evaluate the transition probabilities between these states, and determine the optimal strategies for resource allocation and cooperation. For example, in FSS, MDPs can be used to model scenarios in which satellites dynamically allocate bandwidth based on changing operational demands and state membership.

3.4. FoS case study: Federated satellite systems.

Unlike other distributed satellite systems (DSS), which may involve constellations, clusters or swarms designed for integrated operations under centralized control, FSS involve voluntary cooperation between independently owned spacecraft that collaborate on an opportunistic basis [22]. This distinction differentiates FSS from other DSS architectures by introducing opportunism to resource sharing. Satellites act as both customers and resource providers, depending on their operational status and capabilities, enabling more dynamic and cost-effective missions.

The International Space Station (ISS) has been considered as a provider of satellite resources such as computing power [22] – a concept that was later explored for a cloud computing infrastructure aboard the ISS [23]. In the ISS scenario, the Station contributes resources such as downlink bandwidth, data storage, and computing power to a network of client spacecraft. The study shows that incorporating the ISS into a federated system can help offset the high operational and lifecycle costs of manned space programs, which often exceed hundreds of billions. The federation value assessment synergy combines technical and economic analyses to identify potential customers, rank them by affordability, and assess the financial feasibility of FSS operations. The results show that FSS not only improves the sustainability of space missions but also creates scalable commercial markets for in-orbit resource sharing.

Differing from other distributed satellite architectures, federated satellites also offer advantages in addressing inefficiencies related to underutilized satellite resources. For instance, each satellite experiences idle periods during its operations where available bandwidth or processing power goes unused. FSS enables the sharing of these resources that would be wasted otherwise. This capability has particular impact in orbital environments featuring natural variability of resource supply – such intermittent ground station coverage in Low Earth Orbit (LEO), which often results in latency and resource bottlenecks. By taking advantage of the cooperative structure of the FSS, satellites can cross-link data through peers with more immediate access to ground stations, thereby reducing latency and improving overall mission responsiveness [22].

The distinction between federated satellite systems and other distributed is in their focus on mutual benefits and decentralized control. While constellations optimize global coverage and revisit times through centralized coordination, federations prioritize flexibility and economic efficiency by dynamically matching supply and demand of in-orbit resources. Swarms, which often rely on self-organization and autonomous assignment of tasks, differ from FSS in that they lack explicit economic frameworks for resource sharing [22].

4. A CONCEPTUAL INTRODUCTION TO HETERO-FUNCTIONAL GRAPH THEORY[1]

4.1. Introduction

Hetero-functional Graph Theory (HFGT) leverages MBSE and network science to capture heterogeneous phenomena in heterogeneous systems within a common modeling paradigm [24]. HFGT can be used to model an arbitrary number of networked systems of arbitrary topology connected arbitrarily [24]. More specifically, HFGT utilizes multiple graph-based data structures to support a matrix-based quantitative analysis, inheriting the heterogeneity of conceptual and ontological constructs found in MBSE including system form, system function, and system concept [24]. Furthermore, it can be used to reconcile and subsequently generalize both multi-layer networks [25, 54], axiomatic design models [52, 56], system dynamics models [40], bond graph models, and linear graph models [39]. This ability to reconcile system models from disparate disciplinary sources, as well as the ability to model SoS of arbitrary topology, allows HFGT to conduct novel structural analyses [57, 58], dynamic simulations [59], and optimal decision problems [60].

The modeling process abstracts a “real world” system and represents the abstraction with a model. The model refers to the real-world system, but this reference is always indirect, as an abstraction is always made in the modeling process. Although the abstraction of the real-world system may be either conceptual or linguistic, it is important to distinguish between a conceptual abstraction, residing in the mind, and a linguistic abstraction, residing within a predefined language usually associated ontologically with a real-world knowledge domain. Hetero-functional graph model(s) are the abstracted model of the real-world system, while hetero-functional graph theory provides the means of representing ontologically heterogeneous real-world domain conceptualizations in a common mathematical and computational language.

HFGT has been applied to SoS in numerous application domains individually, including multi-modal electrified transportation systems [50, 65, 66], microgrid-enabled production systems [49], personalized healthcare delivery systems [48, 59, 67], hydrogen-natural gas systems [60], the energy-water-nexus [68], and the American multi-modal energy system [58].

4.2. Instantiated, reference, and meta-architectures for Systems of Systems

To gain a better understanding of the convergence challenge associated with complex SoS, including societal challenges of the Anthropocene, consider what happens when the real-world system is a system of ontologically heterogeneous systems: the real domain is a real-domain-of-real-domains that unites the individual domains into one. For example, the study of the food-energy-water nexus in general (rather than for a specific region) may constitute such a real-domain-of-real-domains. It needs to be abstracted by humans, in the mind, in a domain-conceptualization-of-domain-conceptualizations. Similarly, that must be referred to using a language-of-languages. Several convergence challenges immediately arise. First, humans are typically trained in a single domain conceptualization, rather than multiple knowledge domains. Indeed, it is far from clear that there even exists a single human (let alone many) who has sufficient knowledge of the domain-conceptualization-of-domain-conceptualizations. In the absence of such an individual, a group of individuals –each with their own individual domain conceptualizations– must somehow collaborate to discuss the real-domain-of-real-domains (e.g., the food-energy-water nexus independent of a specific region). They immediately find that each domain-conceptualization comes with its associated language and a language-of-languages emerges. Because each of these languages was developed entirely independently to address the needs of its associated real domain, the language-of-languages is highly divergent and a common, convergent understanding between languages is difficult to achieve. To overcome this impasse, it is possible that the language-of-languages develops a translation capability between each of the languages pertaining to each real-domain. While this strategy is relatively straightforward for only two languages with a single translator, it does not scale when there are *N* real-domains that require *N(N-1)* translators between *N* languages. The only alternative is to invest in the development of a language-of-languages that reconciles the individual languages into a single common language.

HFGT adopts the latter approach, where a single common

[1] *This section abridges several previously published works into a qualitative introduction to hetero-functional graph theory [24, 25].*

language serves as a language-of-languages. The development of a single common language for a domain-conceptualization-of-domain-conceptualizations requires three types of system architectures. As shown in Figure 1, these are the instantiated, reference, and meta-architectures.

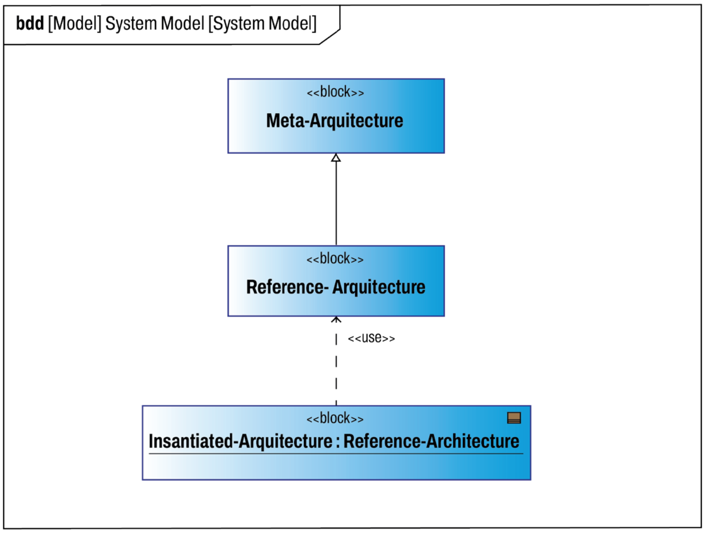

bdd [Model] System Model [System Model]
<<block>>
Meta-Arquitecture
<<block>>
Reference- Arquitecture
<<use>>
<<block>>
Insantiated-Arquitecture : Reference-Architecture

Figure 1. SysML Block Definition Diagram. Systems architecture can be represented at three levels of abstraction with instantiated, reference, and meta-architectures.

Here, we conceive a system architecture as consisting of three parts: the real-world or physical architecture, the functional architecture, and the mapping of the latter onto the former in a system concept (or allocated architecture). The physical architecture is a description of the decomposed elements of the system without any specification of the performance characteristics of the physical resources that comprise each element. The functional architecture is a description of the system processes in a solution-neutral way, structured in serial, or parallel, and potentially in hierarchical arrangements. The system concept as a mapping of the functional architecture onto the physical architecture completes the system architecture. For a hetero-functional graph model to be correctly specified, it is assumed that the entirety of any given process must be completed by a given resource.

An instantiated systems architecture is a case-specific architecture that represents a real-world scenario. At this level, the physical architecture consists of a set of instantiated resources, and the functional architecture consists of a set of instantiated system processes. The mapping in the system concept defines which resources perform what processes.

The reference architecture generalizes instantiated system architectures. Instead of using individual instances as elements of the physical and functional architecture, the reference architecture is expressed in terms of domain-specific classes of these instances. In this way, the reference architecture captures the essence of existing instantiated architectures. It also provides a vision of future needs that can provide guidance for developing new instantiated system architectures. Such a reference architecture facilitates a shared understanding across multiple disciplines or organizations about the current architecture and its future evolution. A reference architecture is based on concepts proven in practice. Most often, preceding architectures are mined for these proven concepts. The reference architecture, therefore, generalizes instantiated system architectures to define an architecture that is generally applicable in a discipline. However, the reference architecture does not generalize beyond the domain conceptualization.

The meta-architecture further generalizes reference architectures. Instead of domain specific elements, it is expressed in terms of *domain-neutral* classes. A reference architecture is composed of “primitive elements” that generalize the domain-specific functional and physical elements into their domain-neutral equivalents. While no single engineering system meta-architecture has been developed for all purposes, several modeling methodologies have been developed that span several discipline-specific domains. In the design of dynamic systems, bond graphs [27-29] and linear graphs [30-34] use generalized capacitors, resistors, inductors, gyrators and transformers as primitive elements. In the system dynamics of business, stocks and flows are often used as primitives [35, 36]. Finally, formal graph theory [37, 38] introduces nodes and edges as primitive elements. Each of these domains has their respective sets of applications. However, their sufficiency must ultimately be tested by an ontological analysis of soundness, completeness, lucidity, and laconicity [24]. Hetero-functional graph theory, as the next section elaborates, utilizes its own meta-architecture that, in recent years, has been shown to generalize linear graphs, bond graphs, system dynamics, and formal graph theory [25, 39, 40]. Given the importance of ontological clarity, HFGT has taken special care in the translation of this meta-architecture from its description in the systems modeling language (SysML) [41-43] to its mathematical representation.

4.3. Essential elements of hetero-functional graph theory

This section introduces the essential elements of hetero-functional graph theory in terms of its underlying meta-architecture. Unlike other meta-architectures, hetero-functional graph theory stems from the universal structure of human language with subjects and predicates and the latter made up of verbs and objects [24, 25]. A real-world engineering system includes a set of resources as subjects, a set of system processes as predicates, and a set of operands as their constituent objects [44-46], where:

- A **system operand** is an asset or object that is operated on or consumed during the execution of a process.
- A **system process** is an activity that transforms or transports a predefined set of input operands into a predefined set of output operands.
- A **system resource** is an asset or object that facilitates the execution of a process. Three types are defined: transformation resources, independent buffers, and transportation resources.

As shown in Fig. 2, these operands, processes, and resources are organized in a meta-architecture [41-43]. Importantly, operands in the engineering system have several types including matter, energy, living organisms, information, and money, which makes HFT intrinsically cyber-physical [24]. Interestingly, it is understood that operands, in general, have some sort of state in time. The evolution of this operand state is described by an operand net as a type of Petri net [46]. The relationships between the different types of system processes and the resources that can execute them is further depicted in the hetero-functional graph theory functional meta-architecture (see Fig. 3).

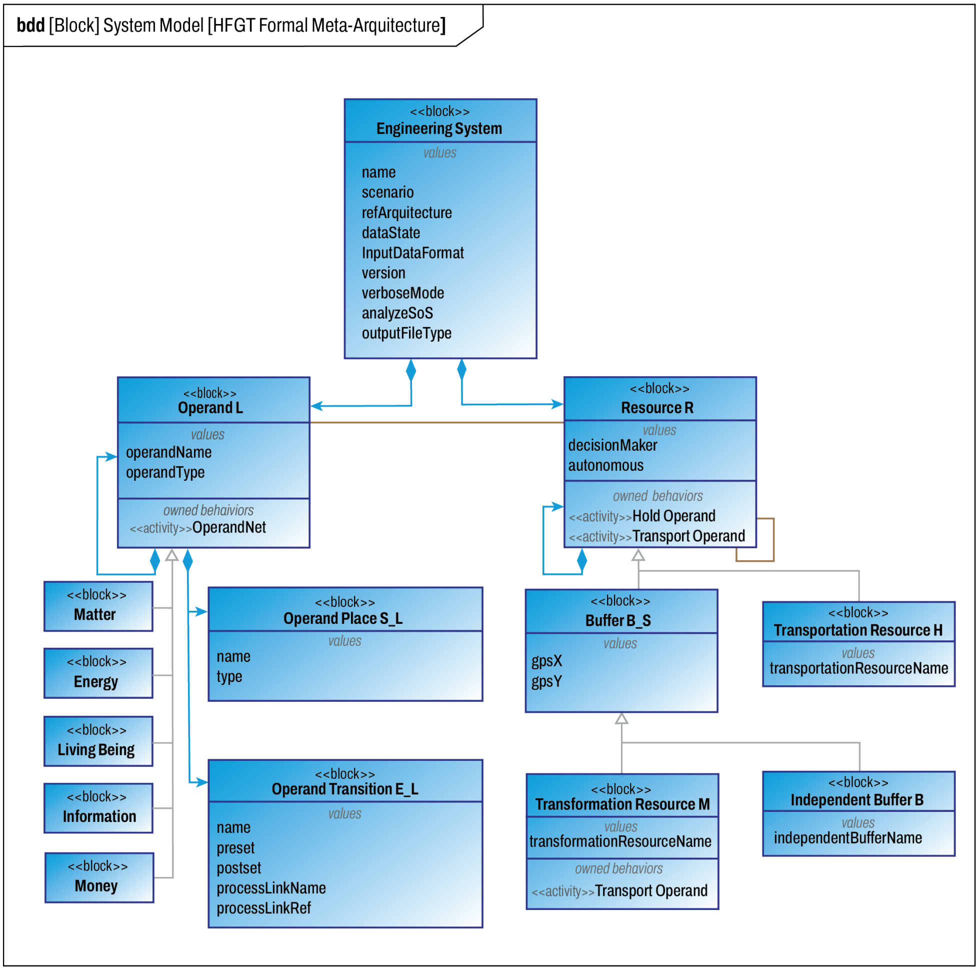


Figure 2. A SysML block diagram of the HFGT formal meta-architecture.

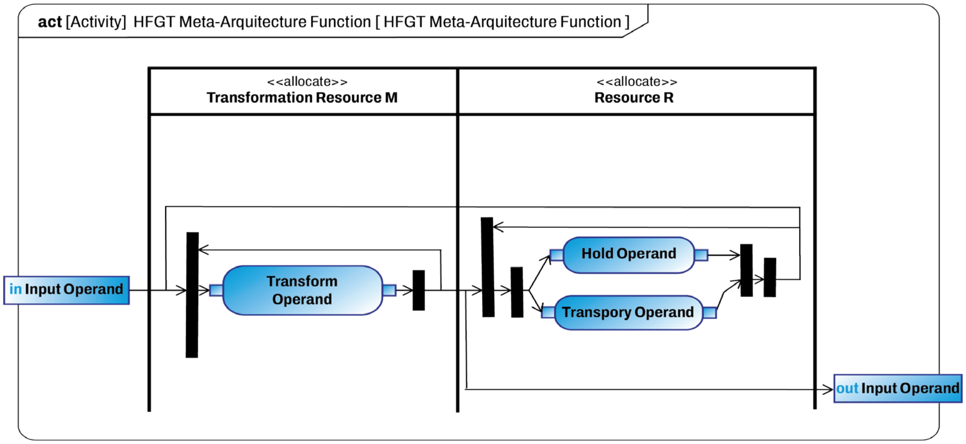


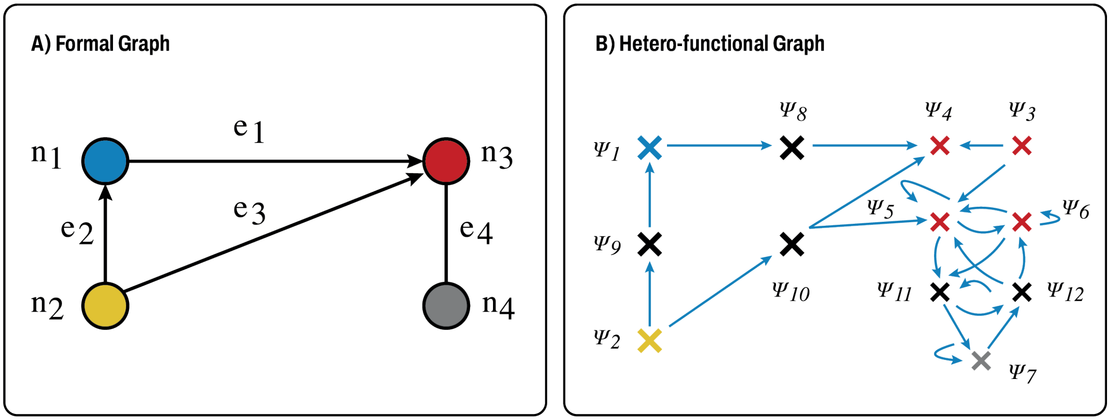
Figure 3. A SysML Activity diagram of the HFGT functional meta-architecture.

Resources are capable of executing one or more system processes to produce a set of capabilities [24]. Intuitively, a capability is articulated as a *subject + verb + operand* sentence of the form <Resource> <executes> <process>. It is important to recognize that, while capabilities are their own distinct entities, they are in reality formed by the allocation of a process to a resource. In Fig. 2, these capabilities appear as owned behaviors of their respective blocks. In Fig. 3, these capabilities appear as actions in their respective swim lanes. At an engineering system level, these allocations are described in the system concept, which may be captured as a binary matrix whose elements are equal to one when an action is available as a system process being executed by a resource. In other words, the system concept forms a bipartite graph between the set of system processes and the set of system resources [51].

Once the engineering system's capabilities have been defined, there is a need to understand how they interact with each other. These appear most clearly as the directed arrows between the actions allocated to swim lanes in Fig. 3. Mathematically, HFGT describes these functional interactions with incidence tensors.

4.4. Illustrative example

This highly abridged conceptual introduction to hetero-functional graph theory can also be explained graphically through the illustrative example shown in Fig. 4.



Legend: Nodes: {n_1-Water Treatment Facility, n_2-Solar PV, n_3-House with Rooftop Solar, n_4-Work Location} Edges: {e_1-Water Pipeline, e_2-Power Line 1, e_3-Power Line, e_4-Road} System Capabilities: {Ψ_1-water treatment facility treats water, Ψ_2-solar PV generates electricity, Ψ_3-house generates electricity, Ψ_4-house consumes water, Ψ_5-house charges EV, Ψ_6-house parks EV, Ψ_7-work location parks EV, Ψ_8-water pipeline transports water from water treatment facility to house, Ψ_9-power line 1 transports electricity from solar PV to water treatment facility, Ψ_{10}-power line 2 transports electricity from solar PV to house, Ψ_{11}-road discharges EV from house to work location, Ψ_{12}-road discharges EV from work location to house}

Figure 4. A visual comparison of a formal graph model and a hetero-functional graph model of the same hypothetical system [24].

Fig. 4 illustrates the difference between a formal graph and a hetero-functional graph (HFG). The formal graph in Fig. 4a shows a system composed of four nodes: a water treatment facility, a solar PV panel, a house with rooftop solar, and a work location. These are connected by four edges: a water pipeline, two power lines and two roads. In contrast, Fig. 4b shows the associated hetero-functional graph. Instead of four nodes that represent point-like facilities, the hetero-functional graph now has 12 nodes that represent the connected system capabilities. The water treatment facility, solar PV panel, and work location appear unchanged between the two graphs because they each have only one capability. In contrast, the house with rooftop solar provides four capabilities in the HFG. The edges in the formal graph now appear as transportation capabilities in the HFG. Finally, the directed edges in the HFG indicate the logical sequences of these capabilities such that if one follows them a "story" of capabilities emerges. For example, the water treatment facility treats water ($\boldsymbol{\Psi}_1$) and then the water pipeline transports the water from the water treatment facility to the house ($\boldsymbol{\Psi}_8$).

5. FEDERATED MODELING AND SIMULATION

5.1. Introduction

Most SoS are too complex for mathematical analysis. Generally, the behavior of the resulting networks of systems depends on their linkages and their environment, where scientific reductionism is incapable of fully defining behavior.

Simulation has become one of the most used analysis tools for large scale systems because it can take randomness into account and address aggregate as well as very detailed models. Furthermore, as computing speed has increased and communication has improved, there has been even more motivation for using computer simulation for larger and larger problems, such as a supply chain. Applying simulation to SoS leads naturally to distributed simulation (referred to as federations), where existing legacy simulation models (referred to as federates) are integrated over a network of multiple computers. The existence of legacy system models is one driving force for federated modeling and simulation. Another is to allow each system (e.g., a supply chain member) to hide any proprietary information in implementation of the individual simulation but still provide enough information to analyze SoS (e.g., supply chain) as a whole.

5.2. Federation of simulation and time synchronization

The design and development of a distributed network of simulations (i.e., federation) is complex and requires expertise in several disciplines including domain experts, system design and specifications, simulation modeling, and distributed computing and networking. Once system models (e.g., discrete event, system dynamics, agent-based, or physics-based model) exist, the time synchronization of multiple system models (i.e., federates) and information exchange among them are important problems.

To illustrate how a federation is constructed and operated given existing federates, a supply chain is used as an example in this section. Figure 5 depicts a federation of supply chain, integrating geographically dispersed federates modeled by commercial simulation software systems (i.e., Arena for the supplier, ProModel for the assembly plant, AutoMod for the transporter).

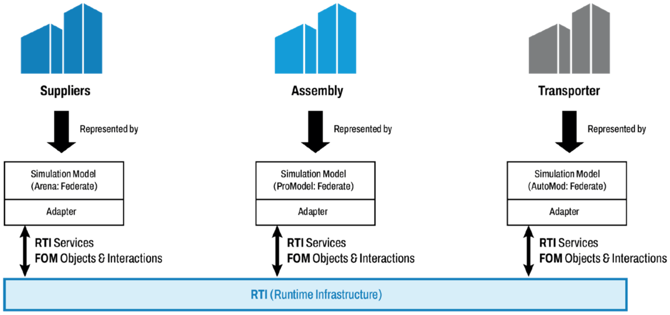

Suppliers
Assembly
Transporter
Represented by
Represented by
Represented by
Simulation Model (Arena: Federate)
Adapter
Simulation Model (ProModel: Federate)
Adapter
Simulation Model (AutoMod: Federate)
Adapter
RTI Services
FOM Objects & Interactions
RTI Services
FOM Objects & Interactions
RTI Services
FOM Objects & Interactions
RTI (Runtime Infrastructure)

Figure 5: Supply chain federation using IEEE Std 1516-2010 framework [69].

Given existing federates, we first need a server (e.g., RTI in Figure 5) which will communicate with the federates. Second, to enable such communications, each federate needs an interface (e.g., Adapter in Figure 5), which will send/receive messages to/from the server (see Figure 6). The major responsibilities of the server are time synchronization and information exchanges among the federates.

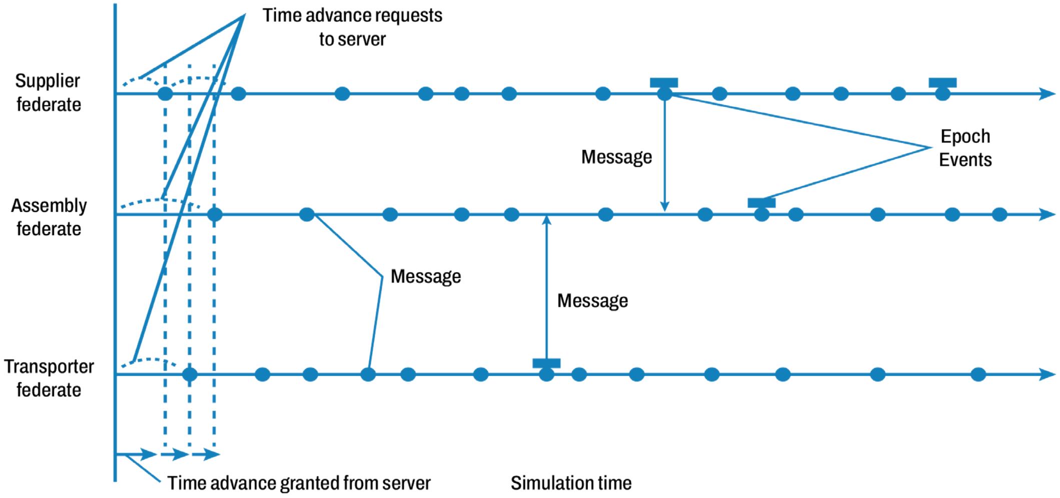


Figure 6: Event-based conservative time synchronization approach [70].

As shown in Figure 6, each federate has two types of events: (1) internal events (continuous models need to be discretized to create internal events), and (2) interaction events. Internal events are the events that do not influence other federates in the federation [70]. An interaction event is defined as an event which influences the behavior of another federate(s). Information concerning interaction events along with the corresponding parameters is communicated to the receiving federates, and the receiving federates adjust their behaviors accordingly in order to enable the system to run properly. If a federate receives an interaction event whose time stamp is earlier than the federate's current simulation clock, then a "causality constraint" violation has occurred; if not reconciled, this violation may invalidate the simulation results. That's why time synchronization is important.

Two major classes of approaches have been used for synchronizing federates – conservative and optimistic. In the conservative time synchronization approach, federates determine the time to their immediate next event, and requests permission from the server to advance to that time. After obtaining requests from all the federates, the server determines the lowest time step requested and grants permission to the federates to move ahead. The federate executes the next event and requests permission to advance to the immediate following event, and the cycle continues. In the conservative approach, no distinction is drawn between the internal events and the interaction events in terms of time management. At each interaction event, a message will be sent from the federate to the server, which then forwards the message to the corresponding federate.

In the optimistic approach, federates step in parallel at fixed or variable increments. If an interaction between federates occurs, they are rolled back to the time where the interaction occurred. However, it is extremely difficult to implement a roll back mechanism in simulation.

5.3. Frameworks enabling federation

A number of frameworks are available to enable distributed simulation in the literature [71-76].

As an example, IEEE Std 1516-2010 describes the framework and rules of the High-Level Architecture (HLA), which is an integrated approach to provide a common architecture for federated simulations. Following a publish and subscribe architecture, HLA can be applicable to various types of operating systems, software, applications, and languages. For example, it allows integration of wide ranges of software: AnyLogic, Simio, Arena, ProModel, Repast, DynusT (traffic simulator), hardware (robots, machines, drones), Unity (game engine), and more.

To maintain or govern models, an open-source Portico (http://porticoproject.org/) or a commercial RTI (e.g., MAK Technologies) can be used, and efforts are needed to develop technical governance. In addition, a governance structure and agreement need to be established among sponsors and users.

6. AGENT BASED MODELING FOR SYSTEMS OF SYSTEMS

6.1. Agent-based modeling paradigm

Agent based modeling is a computational technique used to study and understand the behavior of complex systems. The approach is characterized as bottom-up by modeling the individual entities that make up the system and their interactions. These individual entities referred to as agents can represent individuals, groups, organizations, or any type of autonomous entity. Such models are suited to support the analysis and engineering of SoS and understand their properties as well. ABM gives analyst the flexibility to model key properties of SoS including autonomy of its constituent systems, belonging of its constituent systems to the SoS mission, connectivity among constituent systems, and diverse properties of its constituent systems [77]. The modeling methodology also serves as a digital laboratory for exploring emergent properties and non-intuitive behavior of SoS. This section introduces agent-based modeling basics and discusses the value of ABM for SoS analysis and engineering through some illustrative applications in literature.

Evolution of ABM from theoretical concepts to practical applications spans several decades in various disciplines. The fundamental concept of autonomous agents and their interactions are first proposed by John von Neuman [83], which laid the foundations for John Conway's seminal work on cellular automata, Game of Life, where simple rules applied to cells on a grid lead to emergence of complex behaviors [84]. This work influenced development of early ABMs of social simulations such as Thomas Shelling's segregation model [85], Joshua Epstein and Robert Axtell's Sugarscape model [78], which was an enhanced ABM of Conway's Game of Life and Schelling's segregation model. Other influential models such as Robert Axelrod's work on evolution of cooperation [86] demonstrated the potential strengths of exploring dynamics of agent interactions, emergence, and complexity in social systems and economic systems. The methodology expanded into a wide range of disciplines over the years such as analysis of disease spread, ecosystem dynamics, simulated market dynamics. Computational advances accelerated the use of ABM in recent years. It is now used in a wide range of application domains including various SoS applications such as infrastructure modeling, defense systems analysis, or urban planning. This section first provides a brief snapshot of the modeling paradigm and how it is constructed. It then discusses the value of ABM for SoS analysis and engineering with some ABM examples of SoS engineering problems.

6.2. Modeling agents

While there are different views on the definition of what constitutes an agent, in practice agents have the following features in the context of ABM [80]:

- **Autonomy and self-direction:** The actions of an agent are self-directed and independent, both in its environment and in its interactions with other agents.
- **Identifiable characteristics:** Each agent in the model has identifiable set of characteristics, behavior, and managerial capabilities.
- **Interaction with other agents:** Agents have a set of rules that determine how they interact with other agents.

Other additional properties that may be considered when modeling agents include:

- **Situated in an environment:** Agents may act and interact within an environment where their behavior depends on the interactions with other agents and with the environment.
- **Goal driven behavior:** Agents may have goals, which are decision criteria agents use to assess the effectiveness of their behavior.
- **Ability to learn and adapt:** Agent may have the ability to learn and adapt its behavior based on previous experiences.
- **Resources:** Agents may have resources such as energy, information, or wealth, which dynamically change based on the interactions.

Modeling agents in ABM involves, at a minimum, identifying agent characteristics and identifying agent behaviors. Agent characteristics define specific attributes that distinguish an agent from other types of agents. For example, in a traffic vehicle size/weight is a characteristic that determines if the vehicle is a motorcycle, car, or truck. Other attributes of a vehicle could be speed or fuel consumption. Selection of agent characteristics/features depends on the domain of interest and level of detail necessary to capture the real-world problem. Each agent should also have behavior, which is a set of rules agent acts on when interacting with the environment and with the other agents. An agent's behavior can be simple if-then rules, or it can be described by complex behavioral models such as cognitive decision models or artificial intelligence models. The behavioral models may also incorporate adaptability, where the agent dynamically changes its behavior in response to its experiences. While some of the agent behavior models are based on empirical data or domain knowledge, advanced agent behavior models are often based on theoretical concepts from various disciplines such as game theory, cognitive science, reinforcement learning, and artificial intelligence. Selection of the type of agent behavior model depends on the domain and the ABM's purpose. For example, if the model's purpose is to evaluate the impact of a specific signaling rule on traffic congestion, modeling agent adaptation may not be necessary.

The agent model determines the level of abstraction of the real-world problem as well. While other modeling methodologies are suitable for a specific level of abstraction, ABM provides the flexibility to abstract the real-world system at different levels of detail. Figure 7 compares ABM to other modeling methodologies. The agent model can represent a macro level entity such as an organization vs. a lower-level abstraction such as an engineering design team or at a macro level one can analyze multi-model transportation dynamics by abstracting rail carriers, air cargo freight, and truck carrier behavior vs. lower-level dynamics by modeling individual system dynamics.

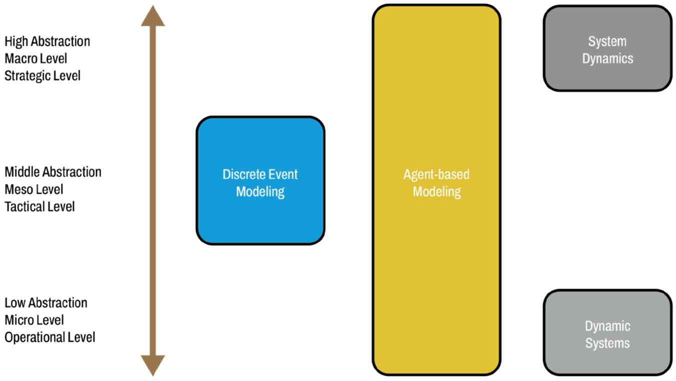


Figure 7. Levels of abstraction of ABM compared to other modeling methods [80].

6.3. Modeling the environment

The environment where agents interact may be modeled in various ways. Figure 8 illustrates the type of environments that can be modeled in ABM. This could be a model of a physical 2-D space, a grid structure, a network, or a complex spatial representation such as a geographic map. It is also possible that agents interact in an aspatial environment model. Regardless of the type of model, environment provides the context for agent interaction.

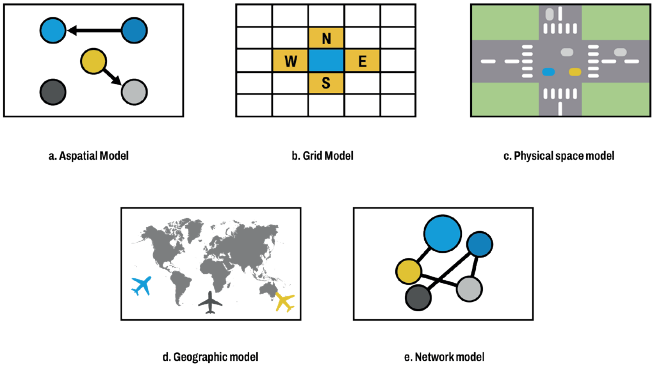


Figure 8. Modeling the environment [80].

6.4. Modeling agent interactions

The interactions between agents can include communication, competition, cooperation, and resource sharing. Regardless of the environment model used to connect the agents, the main point of modeling agent interactions is to identify the rules of local interactions among agents and local resource transfer between agents. This means that agents interact with a limited number of other agents in the total agent population. The nature of these interactions often leads to emergent behaviors.

To give an example of how ABM is constructed, consider Unmanned Air Vehicles (UAV) that are utilized in various SoS missions such as monitoring large areas to provide situational data for surveillance and reconnaissance or locating individuals in distress for search and rescue missions or delivering goods for delivery services [79]. In a search and rescue mission, UAVs form an SoS where each UAV operates autonomously using its sensors and communicates with other UAVs to avoid overlapping search areas. If a UAV detects an individual in distress, it can signal nearby UAVs to send the information to a central command center. An ABM of UAV can be modeled to analyze the effective coverage of the search area, efficient use of the resources, and evaluate the response time of the SoS. The model simulates the behavior and interactions of autonomous UAVs. A description of how ABM may be applied for this SoS mission is provided below:

- **Define types of agents and their characteristics:** In this context, UAVs are the autonomous agents. Several UAV attributes may be considered such as position, velocity, sensor types, or battery level. Mission objectives may distinguish each type of UAV as well. For example, some of the UAVs may be assigned for surveillance, and others for reconnaissance mission.

- **Model agent behavior:** The behaviors for UAV are defined. UAVs sense their environment using their sensors, based on their current state and rules decide which actions to take such as changing direction or changing speed. Finally, they execute their decision and update their position and state. To support sensing and decision making, for example, collision avoidance algorithms and path planning algorithms may be incorporated into the UAV behavior models. Adaptability capabilities can be incorporated into the behavior model as well. For example, UAV may adapt its movement by re-planning its path basedon changes in the environment and mission objectives.

- **Model the environment:** The environment model captures the airspace in which UAVs operate. A geographic map of the area with information on terrain, restricted area zones, or target areas for the UAVs can be modeled to represent the operational environment.

- **Model agent interactions:** Communication protocols to share information among UAVs model the interaction among UAVs. The shared information includes obstacles and UAV status. This information helps UAVs to coordinate movements and avoid collisions. The interaction model may also incorporate task assignment to UAVs based on their current state.

- **Define simulation process:** Initialize the state of the UAVs, which includes initial positions, velocities, and mission objectives as well as setting up the environment model parameters. Then the simulation is executed in discrete time steps. At each time step, each UAV agent senses, decides, and executes a move to update its position and state.

- **Analyze system behavior:** By simulation, one may gain insights into the search and rescue dynamics and possible emergent behaviors that may arise from the interaction of UAVs over the simulation time. These observations provide insights into the dynamics of the system.

To summarize, ABM involves several key steps to create a dynamic simulation that models the complexity of real-world systems:

- Defining types of agents and their characteristics.

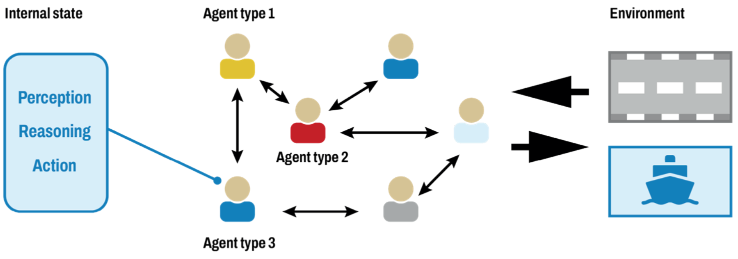
- Modeling agent behavior by defining rules that govern agent behavior and interactions. This could also include adaptation behavior where agents adapt their behavior based on experience and feedback from the environment. Learning algorithms or simple rules to simulate evolving behaviors may be incorporated into agent behavior models.

- Model the environment by selecting a topological model.

- Model agent interactions by defining rules of agent interaction.

- Define simulation time by identifying how time progresses in the simulation. This could be discrete time steps or continuous time.

- Analyze system behavior: Observe and analyze emergent behavior that arise from the interaction of agents over the simulation time. These observations provide insights into the dynamics of the system.



Agent - based Modeling: Agent behavior, agent interaction, environment

Figure 9. Elements of Agent-based Modeling.

6.5. Types of SoS problems suitable for ABM

ABM is suitable for analyzing various types of SoS problems that involve complex interactions and system behaviors. Table 4 provides a sample of SoS domains and problems explored using ABM. Detailed systematic review of SoS problems can be found in [82], where ABM of SoS problems is categorized as SoS-related complex domains, SoS-related social aspects, SoS-related performance issues, SoS-related optimization approaches, SoS simulations for policy issues, SoS engineering-related issues, and SoS theoretical aspects.

Type of SoS problem	Description
SoS-related Complex Domains: Air transportation network	ABM is utilized as a decision support tool. An air transportation network is modeled to investigate how the network performs over time when behavioral patterns of various agents such as airports, and governmental agencies change.
SoS-related Complex Domains: Urban transportation policy analysis	SoS model explores the impact of various urban transportation policies in a city by modeling the behavior of users in urban transportation systems.
SoS-related Social Aspects: UAV-human relationship	SoS model explores the level of autonomy for UAVs utilized for surveillance by analyzing the number of operators, level of autonomy for UAVs, and performance of the SoS.
SoS-related Social Aspects: Smart grid demand response	SoS model is designed where power plants, substations, and consumer agents work together to balance supply and demand. The model provides insights into the dynamics of smart grid demand response.
SoS- related Optimization Approaches: Wildfires	A collaborative SoS model is developed to predict the behavior and effectiveness of various fire-detecting configurations.
SoS related Optimization Approaches: Naval warfare portfolio optimization	Portfolios of SoS architecture configurations are modeled and evaluated against capabilities, costs, and operational risks under a naval warfare scenario. The model analysis support SoS architecture development.
SoS related Performance issues: Network resiliency	A networked naval warfare scenario is modeled to evaluate the resiliency of potential architectures under threats and disruptions.
SoS Simulations for Policy issues: Energy generation	SoS model explores impact of policies for improving energy generation. The model provides insights for system designers to understand alternative SoS design options as well.
SoS Engineering-related issues: SoS engineering communication	A fictious ABM of SoS is developed to analyze the impact of internal knowledge and communication on SoS performance. Results reveal that additional internal knowledge and communication between constituent systems improve the SoS performance.
SoS Engineering-related issues: SoS development based on Wave acquisition process model [70]	An ABM for acknowledged SoS development is modeled based on wave acquisition process model where SoS agent negotiates with individual systems to acquire desired capabilities for the SoS architecture.

Table 4: ABM of SoS problems.

6.6. Value of ABM for SoS engineering

ABM is a valuable tool to support the engineering of SoS as it provides a comprehensive framework to analyze, design, and manage SoS. In a way, ABM serves as a test laboratory to understand how individual system behaviors lead to emergent behavior in a SoS. This is important for managing the complex interactions that are not intuitive from studying the behavior of its individual systems. The simulations may reveal non-intuitive behavior due to complex interactions between its constituent systems which is essential for designing SoS that can adapt and respond to changing conditions. Engineers can also use ABM to simulate various architecture configurations of an SoS to evaluate the impact of architecture alternatives on the overall SoS performance [81]. ABM can be also used as a decision support tool to analyze various scenarios including rare or extreme scenarios. This type of what-if scenario analysis helps to assess the response of SoS mission effectiveness under different conditions, providing valuable insights for systems engineers. Since ABM provides the flexibility to model different agent types and behaviors, it is useful in modeling SoS engineering where individual systems have differing interests and motivations. Thus, ABM provides the flexibility and adaptability to model all types of SoS including directed, acknowledged, collaborative, and virtual. Besides scalable models can be designed by adding new agents to model large scale SoS.

6.7. Challenges

As with any modeling methodology, ABM has its limitations. ABM can be computationally demanding in terms of processing power and memory when simulating large-scale systems with many agents. Modeling agent behavior requires complete and consistent data which may not be available for some application domains. Most importantly, validation of ABM may be challenging as it requires validating the model against empirical data which may not be available for some applications. In addition, the model results are sensitive to model initial conditions and parameters, and stochastic nature of agent interactions make it difficult to reproduce the model results. Despite these challenges, ABM remains a valuable tool for understanding complex systems including SoS.

7. AN SOS APPROACH TO MODELING

7.1. Introduction to SoS modeling

SoS modeling usually involves the integration of multiple system models with multiple levels of abstraction [89, 91]. In contrast to many simpler models of individual systems, SoS models may need to capture processes that operate at different scales (e.g., temporal, spatial, organizational), with exogenous drivers of the individual systems becoming endogenous, and with multiple feedback mechanisms among the individual systems included in the system of systems [89, 91]. In addition, SoS models generally need to integrate knowledge from several disciplines with exchange of information among the disciplines occurring in a coherent and meaningful way. The integration of knowledge is not limited to the technical coupling of the models by disciplinary experts, but to integration among the stakeholders who may be engaged in different systems at different scales. As a result, scale issues [81, 82] are frequently a core consideration of SoS modeling.

7.2. Understanding scale

The range of disciplines involved in SoS modeling (e.g., see [92]) often means that different notions of scale are used in different ways depending on context [89]. The choice of scale clearly needs to be consistent with the *purpose of the modeling*, and with the *spatial* and *temporal* scales represented in the individual systems. The *spatial* and *temporal* features of a system are usually the primary aspects around which scale is considered and framed. These define the time and space of interest including discretization, and the events and processes that are considered important to represent [93]. The spatial scales selected may be influenced by the temporal scales of interest, and vice versa.

Resolution defines the *granularity* of system representation and refers to the unit of spatial/temporal scale represented in each system. Resolution may be spatial or temporal in nature but extends in other ways such as to social systems (e.g., including individuals, groups and communities) and may therefore represent a semantic or conceptual *hierarchy* [85]. Choice of resolution is highly dependent on the modeling context, generally informed by the availability of data, the needs of the model (including for numerical stability, sensitivity and model identifiability), and model purpose [89].

Hierarchy and *levels of organization* relate to the representation of nested relationships among systems [94]. For example, governance systems may co-exist at a range of scales with separate administrative units. Team-based organizations may have hierarchical scales, with members performing a variety of roles within an organization that may be geographically spread across different time zones.

In SoS modeling, each individual model may operate across different spatial/temporal scales, hierarchical levels, and resolutions to incorporate multiple aspects of distinctly separate (disciplinary or sectoral) domains and modeling paradigms (e.g., Bayesian networks, agent-based, and system dynamics [95]).

7.3. Considering scale during the main phases of the modeling process

As shown in Figure 10, the modeling process can be represented as occurring in five main phases [90]. These phases are iterative with activities from multiple phases often occurring concurrently with decisions made in earlier phases being revisited. Modeling practice tends to focus on model formulation and evaluation, but the other phases are equally important if transparency, coherence and equity is required [90].

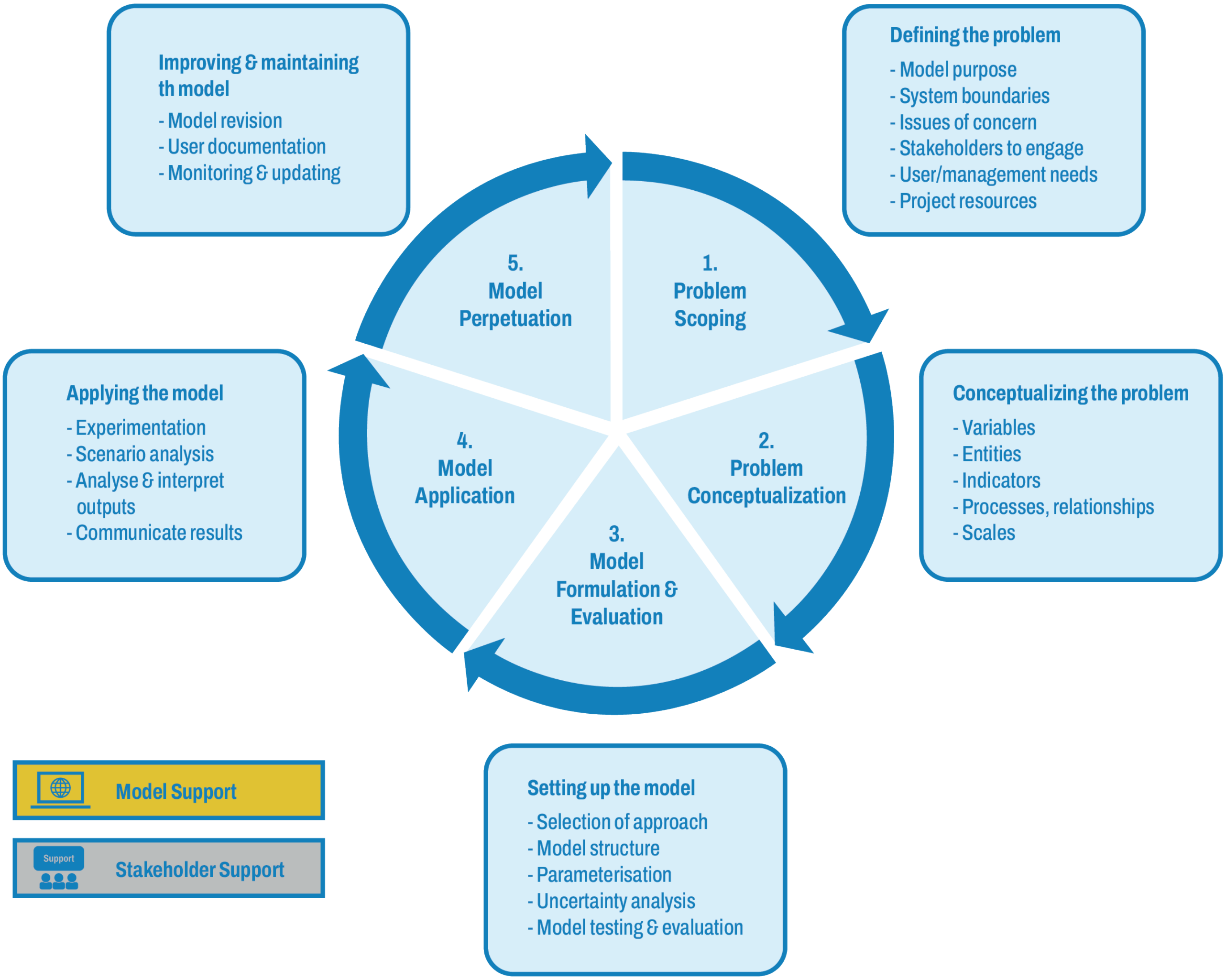


Figure 10: The five main phases of the modeling process with color shading of the iterating circular arrows illustrating the approximate extent of model and stakeholder support in each phase [90].

Each phase requires support from both modeling tools and stakeholder engagement processes, as indicated in Figure 10. Notably, any or all phases could be politically motivated for a specific modeling project, due to, for example, an imbalance of stakeholder representation [96]. This could lead to biases in the model [9], time-place-funding dependent variables [98, 99], or a requirement that the model results should align with specific interests. Thus, appropriate engagement of stakeholders is critical throughout the five phases. Stakeholder engagement ensures the model is fit for purpose, represents multiple perspectives, and is subsequently used and adopted as intended [100].

There are many decision points throughout the lifecycle of the modeling process, including: selecting the boundary of the modeling (i.e., model purpose, problem definition and system boundaries); the evidence base (what data, and whose knowledge or perspective); the model features such as the variables, outputs and scales; the modeling approach; and the model testing and evaluation methods [100]. Different choices at any one of these decision points can result in different modeling pathways, leading to different models and modeling outcomes [101, 102]. This highlights the inherent subjectivity of modeling practices [103], and the need for ongoing reflexivity during the modeling cycle [104].

The sections below are consistent with the five modeling phases [90]. In each section, the first paragraph in each section provides a brief overview of typical actions undertaken in each modeling phase [90]. The remaining paragraphs in each section provide an overview of some considerations related to scale issues [89].

7.4. Problem scoping

The initial problem scoping phase involves defining the problem to be addressed and its scope, including the function or purpose of the modeling, the system boundaries, the issues or questions to be addressed, and the stakeholders to be engaged. This planning phase should also clarify the end-user context covering both user and management needs, problem context including nature of the problem and how well it is understood, and project context which includes resources available such as time, funding, skills and data [100]. This is a critical phase because it determines which interests are addressed and who is allowed to be involved in the formulation of the problem.

While the overarching purpose of the SoS model may be known, the specifics may be less clear at the outset. Development of a consistent and shared view of the scales to be considered involves communication of the scope and interactions across the individual systems. This process can aid in identifying and addressing areas that require reconciliation of different views that often exist among stakeholders. Awareness of the scale issues will likely evolve as the modeling progresses through the iterations. The choice of modeling pathways and methodological framework employed is heavily informed by this awareness [105].

Involvement of stakeholders, including domain experts, through participatory processes can inform the identification of relevant scales in the face of uncertainty and (poor) data availability [106, 107]. Stakeholders can also play a role in selecting and combining data and aid in developing the model purpose.

The purpose and use of individual models may be mismatched if conflicting perspectives over the scope of the modeling are not addressed. Modelers with different goals in mind may only consider scales relevant to their immediate (and often discipline-specific) concerns, leading to an improper selection of individual models. There is potential for a high degree of mismatch between individual models even if modelers coordinate their efforts. Unexpected cascades of effects through scales are commonplace in systems of complex systems [108].

Change in scale may also occur during the modeling process due to new information that triggers a necessary change in model context. The scale of model interactions to be represented can also influence the number and type of individual models included and overall system complexity. The choices regarding scale have implications for how well interactions among systems can be represented with respect to the model purpose. Scope creep, wherein the scale of the modeling is continually extended to cover contexts not originally envisioned, may eventually compromise modeling efforts, as available resources get stretched too thinly to achieve meaningful progress [109].

Choice of scales is further compounded in cases where system boundaries cannot be clearly defined. Coastal zones, atmospheric systems, and natural resource management systems are examples of systems with ambiguous system boundaries. Social systems and their dynamic structures are another example that do not have clear boundaries yet place important constraints on system behavior.

Generally, participatory approaches aim to bring together the multiple goals, issues, and concerns of interest from multiple scales and governance systems by developing a mutually beneficial relationship among stakeholders [110]. Thoughtful consideration of transparency, traceability, and governance issues in engagement and participatory processes [111, 112] will be essential for optimizing saliency, legitimacy, and credibility of the SoS modeling [113].

The participation of a higher diversity of stakeholders in such processes allows for a more holistic representation to be developed, covering potential blind spots in the system conceptualization and avoiding the "siloing" of knowledge [114, 115]. However, including further perspectives may increase the complexity of the modeling and requires careful management of individual expectations and biases [116]. Management of an SoS may at times be predicated on effective management of stakeholders and their level of involvement [94].

Increases in the variety of perspectives also increases potential for conflict between teams, team members, and/or stakeholders. On the one hand, there is evidence that conflict plays a positive role in learning and effective teamwork [117]. Such positive benefits, however, may only occur in cases where there are high levels of pre-existing trust within the group, and when the conflict is task-related rather than interpersonal [118]. Power dynamics within modeling teams and stakeholders therefore need to be considered. Identification and focus on objectives that require participants to work together is an identified foundation towards project success and may additionally help in avoiding conflict [117]. Careful design and management of interactions between teams and stakeholders requires explicit consideration of how the multiple, and at times contradictory, objectives might align or connect. Approaches to conflict resolution and prevention are promising, but still under-utilized techniques. Overall, plans for stakeholder engagement for SoS modeling should explicitly consider the scaling challenges, and devise strategies to deal with these.

7.5. Problem conceptualization

The problem conceptualization phase involves building the evidence base (e.g., expert and stakeholder knowledge, and relevant literature, data, models, and hypotheses) to conceptualize the problem or system, generally in a qualitative sense. This includes identifying key variables, indicators, processes, relationships, entities, and scales, as well as metrics related to model performance [119].

In describing and capturing the essence of the system, development of the conceptual model helps with the design of the subsequent computational model as well as making concrete the model purpose. Two scale-specific aspects to be considered are the approach used for conceptual model development and the formal representation (e.g., equations and technical specifications). The processes that are included or excluded based on perceptions, priorities, beliefs, and values will inevitably influence the data leveraged, the properties of the computational model, and therefore the paths taken.

If differences in conceptual understanding of the scales and their interactions cannot be reconciled, it is possible to create multiple alternative models representing the different hypotheses that can be tested in later stages of the modeling process. Such an approach can also assist in assessing uncertainty rooted in model building choices, as the treatment of scale may affect model outputs and outcomes. Although conceptual diagrams can be developed without specifying the scales involved, explicit consideration of scale is valuable for avoiding misinterpretation of the conceptualization and ensuring key variables and processes are included. A useful exercise, not usually reported but aiding transparency, is to identify what alternative approaches were considered, or could have been considered, and how these may have affected results and outcomes, if adopted.

7.6. Model formulation and evaluation

The model formulation and evaluation phase is typically the main focus of the model development process as it includes the formal description of the model, its implementation in the form of computer software, and the software testing. This phase includes selection of the modeling approach (i.e., the types of individual model used [95]), construction of the model structure, calibration of parameters, uncertainty analysis, and model testing and evaluation.

Transparency in the data collection process and approval from those involved in the modeling are necessary to ensure that collected data remains conceptually relevant across scales. Furthermore, transparency in the context of data collection and usage is a key factor to develop trust among stakeholders and model users, and future adoption of the individual models [120]. Data may need to be transformed to be fully relevant for the context of its intended use, such as up-or-downscaling to ensure compatibility with other processes. Ideally, metadata would include information on the data collection, uncertainty and transformation process, which aids in determining the appropriateness of data for the SoS model. Explicit descriptors of both input and output data can assist in identifying the commensurate level of data collection with respect to available resources.

Modeler bias can have a compounding effect because the choice of data collected, as well as the metadata that describes the data, influences how system interactions are perceived, and thus conceptualized. What may be considered irrelevant in one discipline may dictate modeling pathways in another. In an SoS setting, there are many more participants involved and there is a high degree of uncertainty stemming from the decisions made as a result.

Construction of computational SoS models requires the integration of domain expertise from across the various disciplines involved with technical software development knowledge. While the overarching context may be well-defined within the scoping phase, it is during construction that the individual models, and the scales they represent, are developed, coupled, tested, and validated. Here, existing models may be repurposed or new models developed. The specifics of their initialization, interoperation, method of execution and management of the data involved are to be determined and prototyped in this phase.

Conceptual integration of individual models can benefit from requiring that they be mechanistic as opposed to black boxes. When a model is implemented as a black box, it becomes difficult to evaluate and understand. SoS modeling may make use of pre-existing individual models which constitutes re-purposing, potentially implying the transference of the model assumptions, limitations, and scale to a new context. Model suitability within its original context is not necessarily applicable to the new context. Availability of code alone, for example, does not imply transparency. What is important is the contextual information that is necessary to assess the suitability of the model purpose and functionality.

Technical integration refers to the correctness of model interactions, recognizing the distinction between conceptual or abstract representation and its implementation as software. Successful technical integration of computational models requires the necessary engineering expertise to be available. Crucial considerations are that individual models interact and accordingly that errors will propagate, and that each individual model may undergo its own separate development cycle which invariably necessitates continual adjustments to be made.

Calibration is the process of tuning parameters or altering the functional forms of equations or relations to achieve desired model behavior. In SoS modeling, issues such as non-identifiability and equifinality, curse of dimensionality, computational burden, and data representativeness may all be amplified [90].

Model calibration within the SoS paradigm [81] can take three general approaches: (1) calibration of each individual model independently before integration, (2) calibration of all models together after integration, or (3) a combination thereof. The first approach is the simplest and most straightforward as each individual model would be calibrated within its own domain. While pragmatic, it ignores the effect of representing different scales across the represented SoS and system-system interactions, which in turn affects model behavior and performance of the individual models. The second approach is seemingly the most comprehensive approach to model calibration, as every possible interaction between models could be present in the process of model calibration. Interdisciplinary knowledge is leveraged to ensure calibrated values are both reasonable for the expanded operationalization. The approach, however, has the following major barriers: (1) The search space for model calibration will be excessively large. In addition, new (possibly erroneous) interaction effects might emerge between the parameters of one model with those of another model, especially with different scales of information, which makes the response surface extremely complex for model calibration. The calibration process might then become computationally cumbersome and/or infeasible. (2) The available data with different scales may not be sufficient to properly constrain the model in the process of calibration, as it is not identifiable from the data. There is a risk of overfitting as well, as the available data might be insufficient to produce a generalized model that covers the integrated domain. (3) Expert knowledge for each model may have scale constraints and may not be easily transferable to the full SoS domain. In the third approach, models are integrated one-at-a-time, incrementally adding complexity so that the influence of each individual model can be directly attributed, and subsequent issues can be addressed. While this approach may be as pragmatic as the first, and perhaps as comprehensive as the second, the disadvantage is the time and computational cost to perform sequential coupling and calibration.

SoS models often target large problem domains (e.g., [4]) that necessitate complex models for their assessment and by their nature have a high degree of uncertainty. Quantitative approaches aim to measure the effect of uncertainty in a specific parameter, input, or assumption on an output and allow the numerical characterization of the output distribution and therefore model behavior [121, 122]. Qualitative uncertainty, however, cannot be characterized with a value and arises from sources such as the biases and subjective beliefs of human actors [115]. Qualitative uncertainty can also arise from the modelers' subjective judgment, linguistic imprecision and disagreement among those involved [124, 125].

One commonly suggested approach to restricting model complexity (and possibly runtime) is to screen for insensitive parameters [126]. Such parameters are said to have negligible influence on model output and may be "fixed" or made static in subsequent analyses or otherwise removed from the model. Another is to "tie" related parameters so that they may be represented by a single "hyperparameter" [127]. Reducing the number of parameters, however, does not necessarily equate to a reduction in uncertainty. Rather, it may simply mean that consideration of an uncertainty source is determined to be unimportant for a given context or purpose [126] and doing so may trade off model fidelity under new unseen conditions.

Use of an individual model within an SoS model as opposed to its individual operation, or its modification or simplification through parameter screening and tying, constitutes a change in context. Therefore, parameters initially found to be influential might become inactive and non-influential (and vice versa), or the relationships that led to parameters being tied may change. The change of context also changes the relevance of the assumptions and objectives, and what constitutes an appropriate uncertainty analysis [128]. Uncertainty analysis conducted in one context is not valid across all scales. Thus, premature model simplification may ultimately affect the appropriateness of the SoS model for its overarching purpose. A comprehensive sensitivity analysis under current and possibly alternative conditions can provide valuable insights into a key question*: "when and how does uncertainty matter?*" [129]. An alternate view is that, given the likelihood of limited computational resources, efforts to characterize and communicate uncertainties to stakeholders may be more beneficial than an exhaustive sensitivity analysis [130].

Testing and evaluation can assist in the assessment of the ramifications of scale choice. In this step, reasonableness of model structure and interpretability of relationships within models are assessed along with the traditional analysis of model behavior. Not all outputs produced by the individual models may be relevant for the SoS model purpose and the validity of their outputs is affected due to the integrated nature of SoS modeling. For any evaluation to be effective, the specific model outputs of interest that are relevant for the model purpose must be well understood. Outputs may be at a particular spatio-temporal scale, for instance a long-term average of a model output over a large spatial domain or an extreme event at a specific point location. Issues may also stem from the conceptual suitability of individual models as uncertainty may be propagated throughout and may compound as more models are integrated [131]. Thus, the first step in testing and evaluation involves attempting to refute aspects of SoS model structure and functional relationships within the model based on their lack of correspondence with the represented system and the model outputs. Stakeholders could be leveraged to evaluate the conceptual alignment and appropriateness of the SoS representation at the selected scales.

Evaluation of the behavioral relationships at the integrated SoS level is similar to scientific hypothesis testing or "conceptual testing" [132], wherein functional relationships within the SoS model are examined. Such tests may be especially useful in cases where the internal workings of a model are inaccessible or otherwise unknown but expected behavior of the individual model in the integrated context can be characterized [132]. These approaches can be used to identify impossible or implausible aspects of the SoS model output. If any aspect of model structure or any functional relationship within the model can be shown to be an inadequate representation of the corresponding aspects of the real system, then that portion of the model is refuted [133].

The next step focuses more specifically on the correspondence between model projections and observed data. Strictly speaking, data used in model testing and evaluation must be independent of data used to develop the model [127]. A variety of visual, statistical, and machine learning methods are widely used to evaluate SoS models. The choice of method, however, should be based on the fundamental questions of what scenarios and observations to use in the evaluation. Evaluation of models under the range of conditions similar to those of interest can aid in identifying limitations of the model.

Sensitivity analysis is now regarded as standard practice in modeling [126, 134, 135]. The sensitivity of SoS model behavior to changes to its individual models and their interactions is the target of the assessment. An issue stemming from the likely overparameterization of individual models is equifinality and the lack of identifiability. Equifinality refers to the phenomenon of different implementations or combinations of model structure, parameter values, and their interactions producing equally acceptable results [90]. Identifiability refers to the ability to attribute the influence on model outputs to unique model parameters or structure [136]. Therefore, the greater the number of parameters, the less identifiable the model becomes.

Sensitivities are assessed as part of identifiability analysis, typically by ranking parameters based on their influence on outputs which can aid in determining what parameters require focused efforts to reduce uncertainty or improve identifiability. Information from sensitivity and identifiability analysis can aid in simplifying the model. Naively applying sensitivity and identifiability analysis without consideration of the SoS context may adversely affect modeling outcomes.

Assessment of sensitivities ideally relies on global, rather than local analyses. Use of global sensitivity analyses in model assessment has seen increasing use, despite the lack of uptake or reported use of available software tools to conduct such analyses [137].

7.7. Model application

The model application phase involves experimenting with or running the model using, for example, scenarios of interest, followed by analyzing the model outputs and results. This phase also includes communicating and interpreting model insights to the end users.

A critical aspect in the application of SoS models is that individual models typically evolve independently. Development of each individual model, by necessity, is led by disciplinary experts and undergoes separate, asynchronous, development cycles. As each model may come from different modeling paradigms and sources of knowledge, the implementation may be adjusted over time or even replaced in response to newly acquired knowledge. Advancing towards trial model applications using the expected type and volume of data as early, quickly, and often as possible allows modelers to encounter issues in the model application earlier in the process. Experience gained with each iteration subsequently serves to rectify and protect against future application challenges. Application of the model then requires monitoring and scrutinizing to ensure the underlying models (including their metadata, represented knowledge, and application context) remain current and appropriate.

In cases of long runtime, replacing the most computationally expensive individual models with metamodels may be a viable option. Metamodels approximate the input-output behavior of the original model [138-140] and therefore provide simplified representation(s) of more complex models [141]. Metamodels leverage the emergent simplicity of complex systems and although there are a variety of methods available to accomplish this, generally metamodels require the complex models (i.e., the original individual models) to be available beforehand. Metamodels, being approximations of an original model's response surface, are most relevant to the conditions existing in the datasets upon which they are tuned, so care needs to be taken if using them under conditions that transcend those extant in the data. System forcing data beyond that experienced are of particular concern. If possible, simply allocating more computational resources (e.g., supercomputers) may be the most pragmatic and resource efficient alternative, especially considering the time taken to investigate and implement the options.

In the management context, where SoS models are typically applied, there is a need to adequately describe the level of uncertainties in the SoS model and its predictions. Individual stakeholders may react differently to uncertainties and levels of uncertainty [103]. Presenting scenario results relative to the modeled baseline neatly reduces the inherent biases that come with relying on stakeholder preferences to inform desirable thresholds, as would usually occur in multi-criteria or multi-objective analysis approaches [142, 143]. With such an approach, the acceptability of a (possible) maximum or minimum relative change becomes the focus of stakeholder discussion.

A common requirement shared with tooling for conducting analyses (e.g., for sensitivity and uncertainty analysis, and exploratory modeling) is the provision and definition of parameter values. These may consist of a "default" value, a range within which values may vary, whether these values are categorical, scalar, or regarded as constants (examples may be found in [129, 144, 145]). Categorical values may indicate substitution with other data types or a collection of data types. Such information may be the minimum necessary to conduct appropriate analyses, to reproduce and replicate results, and to support later automation of these activities. Parameter values in effect represent dimensions of scale and the inappropriate selection of their values and ranges may result in misleading results [146, 147].

7.8. Model perpetuation

The final model perpetuation phase is relevant for models that will be used to support ongoing decision making or operational processes to ensure continuous improvement and their long-term adoption. It involves providing documentation to users in running the model and interpreting its outputs, as well as ensuring that plans and mechanisms are in place for appropriately monitoring, maintaining and updating the model. The iterative revision of the model is achieved through ongoing collaboration between modelers and end users.

Where SoS models are used by external stakeholders, some amount of technical support is expected. Without this, use of the model and thus its impact is likely to be minimal. Computational models are software in that they are made of code, and so continued use comes with a baseline cost to cover maintenance, improvements, and updating of documentation. Such capacity is crucial in contexts where long-term management and decision support is an acknowledged requirement. In such cases the design, implementation and documentation of the model should plan for these long-term activities from the beginning. In the SoS context this implies retaining the interdisciplinary knowledge within a team or organization (e.g., [89, 103]).

Documentation is a conduit through which information and knowledge are propagated and provides the necessary context for model evaluation [111]. Without sufficient documentation, it is difficult to understand the context that led to any specific issue, including mismatches between individual models. Lack of context affects the perceived validity of the model conceptualization, restricts model use, rendering the model inappropriate or invalid for its purpose.

The act of documentation itself allows for reflexive and transparent communication and for new insights to be gained. Undocumented assumptions regarding scale and their influence may compromise other individual models, thus holistic awareness of the SoS issues can be obstructed by a lack of documentation. Long-term maintenance and use of the model may also be impeded. No individual holds the knowledge and awareness of the modeling details in their entirety, let alone the effects of interactions among models. It is therefore important to recognize that writing and maintaining documentation should be a team effort, and a culture to support this should be fostered. In practice, there are few incentives for documenting models to such an extent. A key problem in SoS model documentation is that details of the individual models important for the SoS team may be considered unnecessary for the teams developing the individual models. Once again, this stems from potential disconnects between the purpose of the SoS model and the original objectives of each individual model.

Process evaluation in SoS focuses on two facets: achievement of goals and longevity of the models. In terms of goal achievement, process evaluation considers whether the goals of the SoS model were supported by its individual models and, where applicable, whether individual models achieved their own goals. Although satisfying the goals of the individual models may seem an indirect path to satisfying the goals of the SoS model, this interpretation is misleading. An SoS approach to modeling, instead of simply a multi-modeling approach, leverages the autonomy and independence of the individual models. Individual models still need to be capable of yielding their own outcomes, regardless of how those models are used in the context of the SoS model [148].

Evaluation of the longevity of the SoS model, referring to the ability to leverage or reuse the SoS model over time, requires the development and assessment of a targeted plan for its sustainment that includes: (1) monitoring the evolution of the individual models; (2) identifying alternatives for models that may cease their validity, availability or accessibility during the lifetime of the SoS model; (3) establishing a strategy for the continued evolution of the SoS model, including the development of potential transformation frameworks and implementations; and (4) identifying opportunities to facilitate the sustainment of individual systems aligned with the sustainment of the SoS model.

8. CONCLUSIONS

This study employed a range of methodologies to achieve its research objectives. The qualitative analysis provided deep insights into user behaviors and preferences, allowing for a comprehensive understanding of the underlying factors influencing decision-making processes. Quantitative modeling was utilized to predict outcomes with high accuracy, offering a robust framework for analyzing complex data sets. Additionally, case studies were conducted to illustrate practical applications and validate theoretical models, bridging the gap between theory and practice.

The combination of these methodologies enabled a holistic approach to the research, ensuring that findings were both reliable and applicable in real-world scenarios. The qualitative analysis highlighted the importance of context and individual experiences, while the quantitative models offered generalizable results that could be applied across different settings. The case studies provided concrete examples that demonstrated the practical implications of the research, reinforcing the validity of the theoretical constructs.

REFERENCES

1. A.M. Madni and M. Sievers (2018). "Model-based systems engineering: Motivation, current status, and research opportunities." Systems Engineering, vol. 21, no. 3, pp. 172-190. doi: 10.1002/sys.21438

2. M.W. Maier (1998). "Architecting principles for Systems of Systems." Systems Engineering, vol. 1, no. 4, pp. 267-284. doi: 10.1002/(SICI)1520-6858(1998)1:4<267::AID-SYS3>3.0.CO;2-D

3. S.C-Y. Lu, W., Elmaraghy, G. Schuh, and R. Wilhelm (2007). "A Scientific Foundation of Collaborative Engineering," CIRP Annals, vol. 56, no. 2, pp. 605-634. doi: 10.1016/j.cirp.2007.10.010

4. P.D. Collopy and P.M. Hollingsworth (2011). "Value-Driven Design." Journal of Aircraft, vol. 48, no. 3, pp. 749-759. doi: 10.2514/1.C000311

5. P.T. Grogan and A. Valencia-Romero (2019). "Strategic Risk Dominance in Collective Systems Design," Design Science, vol. 5, no. e24. doi: 10.1017/dsj.2019.23

6. M. Maier, "Architecting Principles for Systems of Systems," Systems Engineering, vol. 1, no. 4, p. 267, 1999.

7. A. Golkar and I. Lluch i Cruz, "The Federated Satellite Systems paradigm: Concept and business case evaluation," Acta Astronautica, vol. 111, no. 0, pp. 230-248, 6// 2015, doi: http://dx.doi.org/10.1016/j.actaastro.2015.02.009.

8. U. Pica and A. Golkar, "Sealed-Bid Reverse Auction Pricing Mechanisms for Federated Satellite Systems," Systems Engineering, vol. 20, no. 5, pp. 432-446, 2017, doi: 10.1002/sys.21395.

9. I. Lluch and A. Golkar, "Architecting federations of systems: A framework for capturing synergy," Systems Engineering, vol. 22, no. 4, pp. 295-312, 2019/07/01 2019, doi: https://doi.org/10.1002/sys.21482.

10. I. Lluch, "A framework for architecting federations of engineering systems," PhD, Space Center, Skolkovo Institute of Science and Technology, Moscow, Russia, 2017.

11. A. M. Ross and D. E. Hastings, "The Tradespace Exploration Paradigm," in INCOSE 2005 International Symposium, Rochester, NY, July, 2005 2005.

12. A. M. Ross, D. E. Hastings, J. M. Warmkessel, and N. P. Diller, "Multi-Attribute Tradespace Exploration as Front End for Effective Space System Design," Journal of Spacecraft and Rockets, vol. 41, no. 1, pp. 20-28, 2004/01/01 2004, doi: 10.2514/1.9204.

13. M. Kasunic and W. Anderson, "Measuring systems interoperability: Challenges and opportunities," Software engineering measurement and analysis initiative, 2004.

14. R. Akhtyamov, R. Vingerhoeds, and A. Golkar, "Identifying Retrofitting Opportunities for Federated Satellite Systems," Journal of Spacecraft and Rockets, vol. 56, no. 3, pp. 620-629, 2019, doi: 10.2514/1.A34196.

15. E. Stoll et al., "On-orbit servicing," IEEE Robotics & Automation Magazine, vol. 16, no. 4, pp. 29-33, 2009, doi: 10.1109/MRA.2009.934819.

16. M. E. Sosa, S. D. Eppinger, and C. M. Rowles, “Designing Modular and Integrative Systems,” 2000. [Online]. Available: https://doi.org/10.1115/DETC2000/DTM-14571.

17. D. E. Hastings and C. Joppin, “On-Orbit Upgrade and Repair: The Hubble Space Telescope Example,” Journal of Spacecraft and Rockets, vol. 43, no. 3, pp. 614-625, 2006/05/01 2006, doi: 10.2514/1.15496.

18. B. Ma, Z. Jiang, Y. Liu, and Z. Xie, “Advances in Space Robots for On-Orbit Servicing: A Comprehensive Review,” Advanced Intelligent Systems, vol. 5, no. 8, p. 2200397, 2023/08/01 2023, doi: https://doi.org/10.1002/aisy.202200397.

19. Z. Zheng, S. Xie, H.-N. Dai, X. Chen, and H. Wang, “Blockchain challenges and opportunities: a survey,” International Journal of Web and Grid Services, vol. 14, no. 4, pp. 352-375, 2018/01/01 2018, doi: 10.1504/IJWGS.2018.095647.

20. O. von Maurich and A. Golkar, “Data authentication, integrity and confidentiality mechanisms for federated satellite systems,” Acta Astronautica, vol. 149, pp. 61-76, 2018/08/01/ 2018, doi: https://doi.org/10.1016/j.actaastro.2018.05.003.

21. F. Garcia and E. Rachelson, “Markov Decision Processes,” in Markov Decision Processes in Artificial Intelligence, 2013, pp. 1-38.

22. D. Selva, A. Golkar, O. Korobova, I. L. i. Cruz, P. Collopy, and O. L. de Weck, “Distributed Earth Satellite Systems: What Is Needed to Move Forward?,” Journal of Aerospace Information Systems, vol. 14, no. 8, pp. 412-438, 2017/08/01 2017, doi: 10.2514/1.I010497.

23. S. Briatore, N. Garzaniti, and A. Golkar, “Towards the Internet for Space: bringing cloud computing to space systems,” 36th International Satellite Communications Systems Conference (ICSSC 2018), doi: doi:10.1049/cp.2018.1719

24. Schoonenberg, W.C.H., I.S. Khayal, and A.M. Farid, A Hetero-functional Graph Theory for Modeling Interdependent Smart City Infrastructure. 2019, Berlin, Heidelberg: Springer. 196.

25. Farid, A.M., D.J. Thompson, and W. Schoonenberg, A tensor-based formulation of hetero-functional graph theory. Scientific Reports, 2022. 12(1): p. 18805.

26. Little, J.C., R.O. Kaaronen, J.I. Hukkinen, S. Xiao, T. Sharpee, A.M. Farid, R. Nilchiani, and C.M. Barton, Earth Systems to Anthropocene Systems: An Evolutionary, System-of-Systems, Convergence Paradigm for Interdependent Societal Challenges. Environmental Science & Technology, 2023. 57(14): p. 5504-5520.

27. Brown, F.T., Engineering System Dynamics. 2nd ed. 2007, Boca Raton, FL: CRC Press, Taylor & Francis Group.

28. Karnopp, D., D.L. Margolis, and R.C. Rosenberg, System dynamics: a unified approach. 2nd ed. 1990, New York, NY: Wiley.

29. Paynter, H.M., Analysis and design of engineering systems. 1961: MIT Press.

30. Koenig, H.E., Y. Tokad, and H.K. Kesavan, Analysis of Discrete Physical Systems. 1967: McGraw-Hill.

31. Kuo, B.C., Linear networks and systems. 1967: McGraw-Hill.

32. Blackwell, W.A., Mathematical modeling of physical networks. 1968: Collier-Macmillan.

33. Shearer, J.L., A.T. Murphy, and H.H. Richardson, Introduction to system dynamics. 1967, Reading, UK: Addison-Wesley.

34. Chan, S.-P., S.-Y. Chan, and S.-G. Chan, Analysis of linear Networks and Systems. 1972: Addison-Wesley.

35. Forrester, J.W., Industrial Dynamics – A Major Breakthrough for Decision Makers. Harvard Business Review, 1958. 36(July-August): p. 37-66.

36. Sterman, J.D., Business Dynamics: Systems Thinking and Modeling for A Complex World. Vol. 19. 2000, Boston, MA, USA: Irwin/McGraw-Hill.

37. Newman, M., Networks: An Introduction. 2009, Oxford, UK: Oxford University Press.

38. Van Steen, M., Graph Theory and Complex Networks: An Introduction. 2010: Maarten van Steen.

39. Ghorbanichemazkati, E. and A.M. Farid, Generalizing Linear Graphs and Bond Graph Models with Hetero-functional Graphs for System-of-Systems Engineering Applications. 2025 In review.

40. Naderi, M.M., M.S. Harris, E. Ghorbanichemazkati, J.C. Little, and A.M. Farid, Convergent Anthropocene Systems-of-Systems: Overcoming the Limitations of System Dynamics with Hetero-functional Graph Theory. Journal of Cleaner Production, 2025 In review.

41. Delligatti, L., SysML Distilled - A Brief Guide to the Systems Modeling Language. 2014, Upper Saddle River, NJ: Addison-Wesley.

42. Friedenthal, S., A. Moore, and R. Steiner, A Practical Guide to SysML: The Systems Modeling Language. 2nd ed. 2011, Burlington, MA: Morgan Kaufmann.

43. Weilkiens, T., Systems engineering with SysML/UML modeling, analysis, design. 2007, Burlington, MA: Morgan Kaufmann.

44. Group, S.H.W., Systems Engineering Handbook: A Guide for System Life Cycle Processes and Activities. 2015, International Council on Systems Engineering (INCOSE).

45. Hoyle, D., ISO 9000 pocket guide. 1998, Boston, MA: Butterworth-Heinemann.

46. Girault, C. and R. Valk, Petri nets for systems engineering: a guide to modeling, verification, and applications. 2013: Springer Science & Business Media.

47. Farid, A.M., Reconfigurability Measurement in Automated Manufacturing Systems, in Engineering Department Institute for Manufacturing. 2007, University of Cambridge.

48. Khayal, I.S. and A.M. Farid, Architecting a System Model for Personalized Healthcare Delivery and Managed Individual Health Outcomes. Complexity, 2018. 1(1): p. 1-25.

49. Schoonenberg, W.C.H. and A.M. Farid, A Dynamic Model for the Energy Management of Microgrid-Enabled Production Systems. Journal of Cleaner Production, 2017. 1(1): p. 1-10.

50. Farid, A.M., A Hybrid Dynamic System Model for Multi-Modal Transportation Electrification. IEEE Transactions on Control System Technology, 2016. PP(99): p. 1-12.

51. Farid, A.M., Multi-Agent System Design Principles for Resilient Coordination & Control of Future Power Systems. Intelligent Industrial Systems, 2015. 1(3): p. 255-269.

52. Farid, A.M., An engineering systems introduction to axiomatic design, in Axiomatic Design in Large Systems: Complex Products, Buildings & Manufacturing Systems, A.M. Farid and N.P. Suh, Editors. 2016, Springer: Berlin, Heidelberg. p. 1-47.

53. Viswanath, A., E.E.S. Baca, and A.M. Farid, An Axiomatic Design Approach to Passenger Itinerary Enumeration in Reconfigurable Transportation Systems. IEEE Transactions on Intelligent Transportation Systems, 2014. 15(3): p. 915-924.

54. Thompson, D.J. and A.M. Farid. Reconciling Formal, Multi-Layer, and Hetero-functional Graphs with the Hetero-functional Incidence Tensor. in 2022 17th Annual System of Systems Engineering Conference (SOSE). 2022.

55. Kivelä, M., A. Arenas, M. Barthelemy, J.P. Gleeson, Y. Moreno, and M.A. Porter, Multilayer networks. Journal of complex networks, 2014. 2(3): p. 203-271.

56. Park, G.-J. and A.M. Farid, Design of Large Engineering Systems, in Design Engineering and Science, N.P. Suh, M. Cavique, and J. Foley, Editors. 2021, Springer: Berlin, Heidelberg. p. 367-415.

57. Thompson, D., W.C.H. Schoonenberg, and A.M. Farid, A Hetero-functional Graph Resilience Analysis of the Future American Electric Power System. IEEE Access, 2021. 9: p. 68837-68848.

58. Thompson, D.J. and A.M. Farid, A hetero-functional graph structural analysis of the American Multi-Modal Energy System. Sustainable Energy, Grids and Networks, 2024. 38: p. 101254.

59. Khayal, I.S. and A.M. Farid, A Dynamic System Model for Personalized Healthcare Delivery and Managed Individual Health Outcomes. IEEE Access, 2021. 9: p. 1-16.

60. Schoonenberg, W.C.H. and A.M. Farid, Hetero-functional network minimum cost flow optimization: A hydrogen–natural gas network example. Sustainable Energy, Grids and Networks, 2022. 31: p. 100749.

61. Farid, A.M., Measures of reconfigurability and its key characteristics in intelligent manufacturing systems. Journal of Intelligent Manufacturing, 2017. 28(2): p. 353-369.

62. Farid, A.M. and D.C. McFarlane. A Development of Degrees of Freedom for Manufacturing Systems. in IMS 2006 5th International Symposium on Intelligent Manufacturing Systems: Agents and Virtual Worlds. 2006. Sakarya, Turkey.

63. Farid, A.M. and D.C. McFarlane, Production degrees of freedom as manufacturing system reconfiguration potential measures. Proceedings of the Institution of Mechanical Engineers, Part B, 2008. 222(10): p. 1301-1314.

64. Farid, A.M. and L. Ribeiro, An Axiomatic Design of a Multiagent Reconfigurable Mechatronic System Architecture. IEEE Transactions on Industrial Informatics, 2015. 11(5): p. 1142-1155.

65. Farid, A.M., Electrified transportation system performance: Conventional vs. online electric vehicles, in The On-line Electric Vehicle: Wireless Electric Ground Transportation Systems, N.P. Suh and D.H. Cho, Editors. 2017, Springer: Berlin, Heidelberg. p. 279-313.

66. Van der Wardt, T.J.T. and A.M. Farid A Hybrid Dynamic System Assessment Methodology for Multi-Modal Transportation-Electrification. Energies, 2017. 10, DOI: 10.3390/en10050653.

67. Khayal, I.S. and A.M. Farid, Axiomatic Design Based Volatility Assessment of the Abu Dhabi Healthcare Labor Market. Journal of Enterprise Transformation, 2015. 5(3): p. 162-191.

68. Farid, A.M., A Hetero-functional Graph Resilience Analysis for Convergent Systems-of-Systems. 2025 In review.

69. J Venkateswaran, YJ Son, International Journal of Industrial Engineering. 2004. 11, 151-159

70. Rathore, A., Balaraman, B., Zhao, X., Venkateswaran, J., Son, Y., and Wysk, R. (2005), "Development and Benchmarking of an Epoch Time Synchronization Method for Distributed Simulation," Journal of Manufacturing Systems, Vol. 24, No. 2, pp. 69–78.

71. Salado A, Kannan H., Szajnfarber Z., Rouse W., Son Y., Nirav Merchant, A Reference Architecture for a Policy Test Laboratory.

72. IEEE. (2010). IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA)-- Framework and Rules. IEEE Std 1516-2010 (Revision of IEEE Std 1516-2000), 1-38. doi:10.1109/IEEESTD.2010.5553440

73. Rouse, W. B. (2019). Computing Possible Futures: Model Based Explorations of "What if?". Oxford, UK: Oxford University Press.

74. Rouse, W. B. (2022). Transforming Public-Private Ecosystems: Understanding and Enabling Innovation in Complex Systems. Oxford, UK: Oxford University Press.

75. Singh, R., & Mathirajan, M. (2014, 9-12 Dec. 2014). A conceptual simulation framework for the performance assessment of lot release policies. Paper presented at the 2014 IEEE International Conference on Industrial Engineering and Engineering Management.

76. Hurt, T., McDonnell, J., & McKelvy, T. (2006, 3-6 Dec. 2006). The Modeling Architecture for Technology, Research, and Experimentation. Paper presented at the Proceedings of the 2006 Winter Simulation Conference.

77. Iwanaga, T., H.-H. Wang, S.H. Hamilton, V. Grimm, T.E. Koralewski, A. Salado, S. Elsawah, S. Razavi, J. Yang, P. Glynn, J. Badham, A. Voinov, M. Chen, W.E. Grant, T.R. Peterson, K. Frank, G. Shenk, C.M. Barton, A.J. Jakeman, and J.C. Little, Socio-technical scales in socio-environmental modeling: Managing a system-of-systems modeling approach. Environmental Modelling & Software, 2021. 135: p. 104885.

78. Jakeman, A.J., S. Elsawah, H.-H. Wang, S.H. Hamilton, L. Melsen, and V. Grimm, Towards normalizing good practice across the whole modeling cycle: its instrumentation and future research topics. Socio-Environmental Systems Modelling, 2024. 6: p. 18755.

79. Little, J.C., E.T. Hester, S. Elsawah, G.M. Filz, A. Sandu, C.C. Carey, T. Iwanaga, and A.J. Jakeman, A tiered, system-of-systems modeling framework for resolving complex socio-environmental policy issues. Environmental Modelling & Software, 2019. 112: p. 82-94.

80. Little, J.C., R.O. Kaaronen, J.I. Hukkinen, S. Xiao, T. Sharpee, A.M. Farid, R. Nilchiani, and C.M. Barton, Earth Systems to Anthropocene Systems: An Evolutionary, System-of-Systems, Convergence Paradigm for Interdependent Societal Challenges. Environmental Science & Technology, 2023. 57(14): p. 5504-5520.

81. Cash, D.W., W.N. Adger, F. Berkes, P. Garden, L. Lebel, P. Olsson, L. Pritchard, and O. Young, Scale and Cross-Scale Dynamics: Governance and Information in a Multilevel World. Ecology and Society, 2006. 11(2).

82. Ostrom, E., A diagnostic approach for going beyond panaceas. Proceedings of the National Academy of Sciences, 2007. 104(39): p. 15181-15187.

83. Kelly, R.A., A.J. Jakeman, O. Barreteau, M.E. Borsuk, S. ElSawah, S.H. Hamilton, H.J. Henriksen, S. Kuikka, H.R. Maier, A.E. Rizzoli, H. van Delden, and A.A. Voinov, Selecting among five common modelling approaches for integrated environmental assessment and management. Environmental Modelling & Software, 2013. 47(0): p. 159-181.

84. Macpherson, E., R.I. Cuppari, A. Kagawa-Viviani, H. Brause, W.A. Brewer, W.E. Grant, N.M. Herman-Mercer, B. Livneh, K.R. Neupane, T.N. Petach, C.N. Peters, H.-H. Wang, C. Pahl-Wostl, and H. Wheater, Setting a pluralist agenda for water governance: Why power and scale matter. WIREs Water, 2024. 11(5).

85. Packett, E., N.J. Grigg, J. Wu, S.M. Cuddy, P.J. Wallbrink, and A.J. Jakeman, Mainstreaming gender into water management modelling processes. Environmental Modelling & Software, 2020. 127: p. 104683.

86. Melsen, L.A., It Takes a Village to Run a Model—The Social Practices of Hydrological Modeling. Water Resources Research, 2022. 58(2): p. e2021WR030600.

87. Sanz, D., J. Vos, F. Rambags, J. Hoogesteger, E. Cassiraga, and J.J. Gómez-Alday, The social construction and consequences of groundwater modelling: insight from the Mancha Oriental aquifer, Spain. International Journal of Water Resources Development, 2019. 35(5): p. 808-829.

88. Hamilton, S.H., C.A. Pollino, D.S. Stratford, B. Fu, and A.J. Jakeman, Fit-for-purpose environmental modeling: Targeting the intersection of usability, reliability and feasibility. Environmental Modelling & Software, 2022. 148: p. 105278.

89. Grimm, V. and U. Berger, Robustness analysis: Deconstructing computational models for ecological theory and applications. Ecological Modelling, 2016. 326: p. 162-167.

90. Lahtinen, T.J., J.H.A. Guillaume, and R.P. Hämäläinen, Why pay attention to paths in the practice of environmental modelling? Environmental Modelling & Software, 2017. 92: p. 74-81.

91. Voinov, A., R. Seppelt, S. Reis, J.E.M.S. Nabel, and S. Shokravi, Values in socio-environmental modelling: Persuasion for action or excuse for inaction. Environmental Modelling & Software, 2014. 53: p. 207-212.

92. Zare, F., J.H.A. Guillaume, A.J. Jakeman, and O. Torabi, Reflective communication to improve problem-solving pathways: Key issues illustrated for an integrated environmental modelling case study. Environmental Modelling & Software, 2020. 126: p. 104645.

93. MacLeod, M. and M. Nagatsu, What does interdisciplinarity look like in practice: Mapping interdisciplinarity and its limits in the environmental sciences. Studies in History and Philosophy of Science Part A, 2018. 67: p. 74-84.

94. Hamilton, S.H., S. ElSawah, J.H.A. Guillaume, A.J. Jakeman, and S.A. Pierce, Integrated assessment and modelling: Overview and synthesis of salient dimensions. Environmental Modelling & Software, 2015. 64: p. 215-229.

95. Kragt, M.E., B.J. Robson, and C.J.A. Macleod, Modellers' roles in structuring integrative research projects. Environmental Modelling & Software, 2013. 39: p. 322-330.

96. Tranquillo, J., An Introduction to Complex Systems: Making Sense of a Changing World. 2019: Springer Nature.

97. Sarosa, S. and A. Tatnall, Failure to Launch: Scope Creep and Other Causes of Failure from an Actor-Network Theory Perspective. International Journal of Actor-Network Theory and Technological Innovation (IJANTTI), 2015. 7(4): p. 1-13.

98. Thompson, J.L., Building Collective Communication Competence in Interdisciplinary Research Teams. Journal of Applied Communication Research, 2009. 37(3): p. 278-297.

99. Cockerill, K., P. Glynn, I. Chabay, M. Farooque, R.P. Hämäläinen, B. Miyamoto, and P. McKay, Records of engagement and decision making for environmental and socio-ecological challenges. EURO Journal on decision processes, 2019. 7(3-4): p. 243-265.

100. Glynn, P.D., A.A. Voinov, C.D. Shapiro, and P.A. White, From data to decisions: Processing information, biases, and beliefs for improved management of natural resources and environments. Earth's Future, 2017. 5(4): p. 356-378.

101. Cash, D.W., W.C. Clark, F. Alcock, N.M. Dickson, N. Eckley, D.H. Guston, J. Jäger, and R.B. Mitchell, Knowledge systems for sustainable development. Proceedings of the National Academy of Sciences, 2003. 100(14): p. 8086-8091.

102. Hoekstra, A., B. Chopard, and P. Coveney, Multiscale modelling and simulation: a position paper. Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences, 2014. 372(2021).

103. Hoekstra, A.G., S. Portegies Zwart, and P.V. Coveney, Multiscale modelling, simulation and computing: from the desktop to the exascale. Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences, 2019. 377(2142): p. 20180355.

104. Martin, D.M., S.J. Powell, J.A. Webb, S.J. Nichols, and N.L. Poff, An Objective Method to Prioritize Socio-Environmental Water Management Tradeoffs Using Multi-Criteria Decision Analysis. River Research and Applications, 2017. 33(4): p. 586-596.

105. Tjosvold, D., C. Hui, D.Z. Ding, and J. Hu, Conflict values and team relationships: conflict's contribution to team effectiveness and citizenship in China. Journal of Organizational Behavior, 2003. 24(1): p. 69-88.

106. De Dreu, C.K.W., The virtue and vice of workplace conflict: food for (pessimistic) thought. Journal of Organizational Behavior, 2008. 29(1): p. 5-18.

107. Bennett, N.D., B.F.W. Croke, G. Guariso, J.H.A. Guillaume, S.H. Hamilton, A.J. Jakeman, S. Marsili-Libelli, L.T.H. Newham, J.P. Norton, C. Perrin, S.A. Pierce, B. Robson, R. Seppelt, A.A. Voinov, B.D. Fath, and V. Andreassian, Characterising performance of environmental models. Environmental Modelling & Software, 2013. 40(Supplement C): p. 1-20.

108. Barba, L.A., Praxis of Reproducible Computational Science. Computing in Science & Engineering, 2019. 21(1): p. 73-78.

109. Saltelli, A., K. Aleksankina, W. Becker, P. Fennell, F. Ferretti, N. Holst, S. Li, and Q. Wu, Why so many published sensitivity analyses are false: A systematic review of sensitivity analysis practices. Environmental Modelling & Software, 2019. 114: p. 29-39.

110. Zimmermann, H.J., An application-oriented view of modeling uncertainty. European Journal of Operational Research, 2000. 122(2): p. 190-198.

111. Chen, C.-F., H.-w. Ma, and K.H. Reckhow, Assessment of water quality management with a systematic qualitative uncertainty analysis. Science of The Total Environment, 2007. 374(1): p. 13-25.

112. Linkov, I. and D. Burmistrov, Model Uncertainty and Choices Made by Modelers: Lessons Learned from the International Atomic Energy Agency Model Intercomparisons. Risk Analysis, 2003. 23(6): p. 1297-1308.

113. Refsgaard, J.C., J.P. van der Sluijs, A.L. Højberg, and P.A. Vanrolleghem, Uncertainty in the environmental modelling process – A framework and guidance. Environmental Modelling & Software, 2007. 22(11): p. 1543-1556.

114. Pianosi, F., K. Beven, J. Freer, J.W. Hall, J. Rougier, D.B. Stephenson, and T. Wagener, Sensitivity analysis of environmental models: A systematic review with practical workflow. Environmental Modelling & Software, 2016. 79: p. 214-232.

115. Raick, C., K. Soetaert, and M. Grégoire, Model complexity and performance: How far can we simplify? Progress in Oceanography, 2006. 70(1): p. 27-57.

116. Song, X., J. Zhang, C. Zhan, Y. Xuan, M. Ye, and C. Xu, Global sensitivity analysis in hydrological modeling: Review of concepts, methods, theoretical framework, and applications. Journal of Hydrology, 2015. 523: p. 739-757.

117. Razavi, S., R. Sheikholeslami, H.V. Gupta, and A. Haghnegahdar, VARS-TOOL: A toolbox for comprehensive, efficient, and robust sensitivity and uncertainty analysis. Environmental Modelling & Software, 2019. 112: p. 95-107.

118. Reichert, P., Towards a comprehensive uncertainty assessment in environmental research and decision support. Water Science and Technology, 2020. 81(8): p. 1588-1596.

119. Dunford, R., P.A. Harrison, and M.D.A. Rounsevell, Exploring scenario and model uncertainty in cross-sectoral integrated assessment approaches to climate change impacts. Climatic Change, 2015. 132(3): p. 417-432.

120. Iwanaga, T., D. Partington, J. Ticehurst, B.F.W. Croke, and A.J. Jakeman, A socio-environmental model for exploring sustainable water management futures: Participatory and collaborative modelling in the Lower Campaspe catchment. Journal of Hydrology: Regional Studies, 2020. 28: p. 100669.

121. Li, G., Z. Bie, Y. Kou, J. Jiang, and M. Bettinelli, Reliability evaluation of integrated energy systems based on smart agent communication. Applied Energy, 2016. 167: p. 397-406.

122. Norton, J., An introduction to sensitivity assessment of simulation models. Environmental Modelling & Software, 2015. 69: p. 166-174.

123. Razavi, S. and H.V. Gupta, What do we mean by sensitivity analysis? The need for comprehensive characterization of "global" sensitivity in Earth and Environmental systems models. Water Resources Research, 2015. 51(5): p. 3070-3092.

124. Guillaume, J.H.A., J.D. Jakeman, S. Marsili-Libelli, M. Asher, P. Brunner, B. Croke, M.C. Hill, A.J. Jakeman, K.J. Keesman, S. Razavi, and J.D. Stigter, Introductory overview of identifiability analysis: A guide to evaluating whether you have the right type of data for your modeling purpose. Environmental Modelling & Software, 2019. 119: p. 418-432.

125. Douglas-Smith, D., T. Iwanaga, B.F.W. Croke, and A.J. Jakeman, Certain trends in uncertainty and sensitivity analysis: An overview of software tools and techniques. Environmental Modelling & Software, 2020. 124: p. 104588.

126. Castelletti, A., S. Galelli, M. Ratto, R. Soncini-Sessa, and P.C. Young, A general framework for Dynamic Emulation Modelling in environmental problems. Environmental Modelling & Software, 2012. 34: p. 5-18.

127. Christelis, V. and A.G. Hughes, Metamodel-assisted analysis of an integrated model composition: An example using linked surface water – groundwater models. Environmental Modelling & Software, 2018. 107: p. 298-306.

128. Pietzsch, B., S. Fiedler, K.G. Mertens, M. Richter, C. Scherer, eacute, dric, K. Widyastuti, M.-C. Wimmler, L. Zakharova, and U. Berger, Metamodels for Evaluating, Calibrating and Applying Agent-Based Models: A Review. Journal of Artificial Societies and Social Simulation, 2020. 23(2): p. 9.

129. Asher, M.J., B.F.W. Croke, A.J. Jakeman, and L.J.M. Peeters, A review of surrogate models and their application to groundwater modeling. Water Resources Research, 2015. 51(8): p. 5957-5973.

130. Maier, H.R., J.H.A. Guillaume, H. van Delden, G.A. Riddell, M. Haasnoot, and J.H. Kwakkel, An uncertain future, deep uncertainty, scenarios, robustness and adaptation: How do they fit together? Environmental Modelling & Software, 2016. 81: p. 154-164.

131. Reichert, P. and M.E. Borsuk, Does high forecast uncertainty preclude effective decision support? Environmental Modelling & Software, 2005. 20(8): p. 991-1001.

132. Kwakkel, J.H., The Exploratory Modeling Workbench: An open source toolkit for exploratory modeling, scenario discovery, and (multi-objective) robust decision making. Environmental Modelling & Software, 2017. 96: p. 239-250.

133. Pianosi, F., F. Sarrazin, and T. Wagener, A Matlab toolbox for Global Sensitivity Analysis. Environmental Modelling & Software, 2015. 70: p. 80-85.

134. Shin, M.-J., J.H.A. Guillaume, B.F.W. Croke, and A.J. Jakeman, Addressing ten questions about conceptual rainfall–runoff models with global sensitivity analyses in R. Journal of Hydrology, 2013. 503: p. 135-152.

135. Wagener, T. and F. Pianosi, What has Global Sensitivity Analysis ever done for us? A systematic review to support scientific advancement and to inform policy-making in earth system modelling. Earth-Science Reviews, 2019. 194: p. 1-18.

136. Salado, A. Abandonment: A natural consequence of autonomy and belonging in Systems of Systems. in 2015 10th System of Systems Engineering Conference (SoSE). 2015.

137. Baldwin C., Sauser B., and Cloutier R., "Simulation Approaches for System of Systems: Event-based versus Agent Based Modeling, Procedia Computer Science, 44, pp. 363-372, 2015.

138. Acheson P., Dagli C., and Kilicay-Ergin N., "Model Based Systems Engineering for System of Systems Using Agent-Based Modeling" Procedia Computer Science, 16, pp. 11-19, 2013.

139. Wei Y., and Madey G., "Agent-based Simulation for UAV Swarm Mission Planning and Execution", Proceedings of the Agent-Directed Simulation Symposium, Society for Computer Simulation International, 2013.

140. Macal C., and North M., "Agent-based Modeling and Simulation" Proceedings of the 2009 Winter Simulation Conference, pp. 86-98, 2009.

141. Mour A., Kenley R., Davendralingam N., and DeLaurentis D., "Agent-based Modeling for Systems of Systems" INCOSE International Symposium, pp. 973-976

142. Silva R., and Braga R. "Simulating System-of-Systems with Agent-based Modeling: A Systematic Literature Review" IEEE Systems Journal, Vol. 14, No. 3, pp. 3609-3617, 2020.

143. Von Neumann, John. Theory of Self-Reproducing Automata. Ed. Arthur W. Burks. Urbana: University of Illinois Press, 1966.

144. Games, M., (1970) The fantastic combinations of John Conway's new solitaire game "life" by Martin Gardner. Scientific American, 223, pp.120–123.

145. Schelling, T. C. (1971). "Dynamic models of segregation". The Journal of Mathematical Sociology. 1 (2), pp. 143- 186.

146. Epstein J., and Axtell R., Growing Artificial Societies: Social Science from the Bottom Up, Brookings Institution Press, 1996.

147. Axelrod R., The Evolution of Cooperation, New York: Basic Books, 1984.

148. Borshchev A. and Filippov A., "From System Dynamics and Discrete Event to Practical Agent Based Modeling: Reasons, Techniques, Tools" The 22nd International Conference of the System Dynamics Society, July 25-29, 2004, Oxford, England.

JOSÉ LUIS DE ROSARIO SÁNCHEZ-SIMÓN

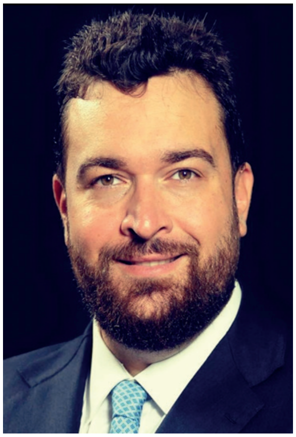
José Luis de Rosario Sánchez-Simón is a Mining Engineer specializing in Energy and Fuels from the Universidad Politécnica de Madrid. He also holds an Executive MBA from Instituto de Empresa. Currently, he is an adjunct professor at Universidad Europea de Madrid, teaching Operations Management in the Master's programs for Industrial Engineering and Engineering Organization, Project Management, and Business.

He works also at Isdefe as a Systems Engineer specializing in strategy, process improvement, and quality assurance systems. He was the winner of the R&D&I competition for his balanced scorecard project. Previously, he worked at ING as a business analyst responsible for Spain operations reporting. At IBM, he was a finance controller for France and Italy, managing revenue, expenditure, and profit analysis for each business line. At Fundación ONCE, he led quality assurance. He implemented RFID-based warehouse management systems at LTR. At STEF He was the quality assurance and logistics specialist for Procter & Gamble in Madrid, Barcelona, Lisbon, and Canary Islands platforms. He is also committed to supporting people with disabilities, sharing his experience through talks at hospitals, universities, and associations.

DR. PAUL GROGAN

Dr. Paul Grogan is an associate professor with the School of Computing and Augmented Intelligence at Arizona State University. He holds a Ph.D. in engineering systems and a S.M. degree in aeronautics and astronautics from the Massachusetts Institute of Technology and a B.S. degree in engineering mechanics and astronautics from the University of Wisconsin. He leads the Collective Design Lab, which develops and studies the use of information-based methods and tools for engineering design of Earth and space systems having distributed or decentralized architectures.

DR. ALESSANDRO GOLKAR



Dr. Alessandro Golkar is a Professor at the Technical University of Munich, Chair of Spacecraft Systems, within the Department of Aerospace Engineering and Geodesy. Dr. Golkar focuses on space systems engineering, technology management, and space entrepreneurship. He previously covered roles across industry and academia including as Vice President in the Technology Planning and Roadmapping organization of the AIRBUS CTO (Toulouse, France), and Space Center Director at Skoltech (Moscow, Russia). He is the lead author of the New Space Economy online course of MIT Professional Education in the US, where he serves as a Visiting Senior Instructor. Dr. Golkar holds a Ph.D. in Aeronautics and Astronautics from MIT, an Executive MBA from Quantic Institute of Technology, and Master's and Bachelor's degrees in aerospace engineering from the University of Rome “La Sapienza”.

DR. AMRO M. FARID

Dr. Amro M. Farid is the Alexander Crombie Humphreys Chair Professor in Economics of Engineering at the Department of Systems and Enterprises at the Stevens Institute of Technology. He is the Principal Systems Scientist at National Energy Analysis Centre at CSIRO – Australia's National Science Agency. He is also a Visiting Scientist at MIT Mechanical Engineering and CEO of Engineering Systems Analytics LLC. At Stevens, he leads the Laboratory for Intelligent Integrated Networks of Engineering Systems (LIINES) and has authored over 165 peer reviewed publications in Smart Power Grids, Hydrogen-Energy-Water Nexus, Electrified Transportation Systems, Industrial Production & Supply Chain Energy Management, Smart Cities, Regions, & Nations

DR. YOUNG-JUN SON

Dr. Young-Jun Son is the James J. Solberg Head and Ransburg Professor of Edwardson School of Industrial Engineering at Purdue University. Prior to this position, he was the Department Head and Professor of Systems and Industrial Engineering at the University of Arizona. His research focuses on a data-driven, multi-scale, simulation and decision model needed for design and control in various applications, including extended manufacturing enterprise, renewable energy and storage network, homeland security, transportation, and social network. He has authored/co-authored over 110 journal papers and 100 conference papers. He is a Fellow of the Institute of Industrial and Systems Engineers (IISE), and has received the Society of Manufacturing Engineers (SME) 2004 Outstanding Young ME Award, the IIE 2005 Outstanding Young IE Award, the IISE Annual Meeting Best Paper Awards (2005, 2008, 2009, 2016, 2018, 2019), and Best Paper of the Year Award in 2007 from International Journal of IE. His research activities have been funded by NSF, AFOSR, DOT/FHWA, US Department of Energy/AZ Commerce Authority, USDA, NIST, Sandia National Lab, Science Foundation of Arizona, Boeing, Samsung, Motorola, Raytheon, Tucson Electric Power, Microsoft, and several application software companies. He is a Department Editor for IISE Transactions, on the editorial board for seven additional journals. He was the vice chair and secretary for the SISO Core Manufacturing Simulation Data (CMSD) Standard Product Development Group. He currently serves on 1) the board of Winter Simulation Conference, and has served on 2) the board of IISE as the Vice President of Continuing Education and 3) as a member of INFORMS Meetings Committee. He has served as co-Program Chair for ISERC 2007, the General Chair for INFORMS Annual Meeting 2018, and the General Chair for Winter Simulation Conference 2019.

DR. NIL H. ERGIN

Dr. Nil H. Ergin is the Professor-in-charge of the Systems Engineering and Engineering Management Programs at the Pennsylvania State University and Associate Professor of Systems Engineering at Penn State Great Valley School of Graduate Professional Studies. She earned her Ph.D. in systems engineering, M.S. in engineering management from the University of Missouri-Rolla (currently known as Missouri University of Science & Technology), and B.S. degree in environmental engineering from Istanbul Technical University. Prior to joining Penn State University, Dr. Ergin worked within the Research Institute for Manufacturing and Engineering Systems (RIMES) at the University of Texas at El Paso where she served on industry funded research contracts and taught for the systems engineering graduate program. Dr. Ergin's research integrates key principles from systems science, problem-solving theories, multi-agent models, and knowledge models to the analysis of wicked problems in applications of system of systems and complex adaptive systems. She is a senior member of IEEE and member of INCOSE.

DR. JOHN LITTLE

Dr. John Little received a BS in Chemical Engineering from the University of Cape Town and an MS and PhD in Environmental Engineering from the University of California, Berkeley. After completing a Postdoc at Lawrence Berkeley National Laboratory, he joined the Department of Civil and Environmental Engineering at Virginia Tech, and is currently Charles E. Via, Jr. Professor. John's research previously focused on process dynamics in environmental systems (building and indoor environment, and water quality in lakes and reservoirs) but has now broadened to process dynamics in Anthropocene systems. Because Anthropocene systems are highly interdependent and dynamically evolving, often with accelerating rates of cultural and technological evolution, the ensuing family of societal challenges (e.g., climate change, renewable energy, adaptive infrastructure, disasters, pandemics, food insecurity, biodiversity loss, sustainable development, resilience and equity) are also highly interdependent and need to be framed and addressed in a holistic fashion. To catalyze the required societal transformations at local, urban, regional and global scales, an evolutionary, system-of-systems convergence paradigm is needed. Dr. Little received a National Science Foundation Career Award in 1996, was elected to the International Society of Indoor Air Quality and Climate Academy of Fellows in 2008, received the Association of Environmental Engineering and Science Professors Outstanding Doctoral Dissertation Award in 2011, and the North American Lake Management Society Technical Merit Research Award in 2014. Dr. Little has been a visiting professor at University of Sydney, Australia; Swiss Federal Institute for Aquatic Science and Technology (Eawag), Switzerland; Tsinghua University, China; National Cheng Kung University, Taiwan; University of Granada, Spain; Centre Scientifique et Technique du Bâtiment, France; and University of La Rochelle, France.

Isdefe
C/ Beatriz de Bobadilla, 3
28040 Madrid
Tel.: +34 91 411 50 11
Email: general@isdefe.es
www.isdefe.es

Isdefe

CUADERNOS DE
Isdefe