The Systems Engineering journal is intended to be a primary source of multidisciplinary information for the systems engineering and management of products and services, and processes of all types. Systems engineering activities involve the technologies and system management approaches needed for:

- Definition of systems, including identification of user requirements and technological specifications;
- Development of systems, including conceptual architectures, tradeoff of design concepts, configuration management during system development, integration of new systems with legacy systems, integrated product and process development; and
- Deployment of systems, including operational test and evaluation, maintenance over an extended life cycle, and re-engineering.

Systems Engineering is the archival journal of, and exists to serve the following objectives of, the International Council on Systems Engineering (INCOSE):

- To provide a focal point for dissemination of systems engineering knowledge;
- To promote collaboration in systems engineering education and research;
- To encourage and assure establishment of professional standards for integrity in the practice of systems engineering;
- To improve the professional status of all those engaged in the practice of systems engineering;
- To encourage governmental and industrial support for research and educational programs that will improve the systems engineering process and its practice.

The journal supports these goals by providing a continuing, respected publication of peer-reviewed results from research and development in the area of systems engineering. Systems engineering is defined broadly in this context as an interdisciplinary approach and means to enable the realization of successful systems that are of high quality, cost-effective, and trustworthy in meeting customer requirements.

The Systems Engineering journal is dedicated to all aspects of the engineering of systems: technical, management, economic, and social. It focuses on the life cycle processes needed to create trustworthy and high-quality systems. It will also emphasize the systems management efforts needed to define, develop, and deploy trustworthy and high quality processes for the production of systems. Within this, Systems Engineering is especially concerned with evaluation of the efficiency and effectiveness of systems management, technical direction, and integration of systems. Systems Engineering is also very concerned with the engineering of systems that support sustainable development. Modern systems, including both products and services, are often very knowledge-intensive, and are found in both the public and private sectors.

The journal emphasizes strategic and program management of these, and the information and knowledge base for knowledge principles, knowledge practices, and knowledge perspectives for the engineering of systems. Definitive case studies involving systems engineering practice are especially welcome.

The journal is a primary source of information for the systems engineering of products and services that are generally large in scale, scope, and complexity. Systems Engineering will be especially concerned with process- or product-line-related efforts needed to produce products that are trustworthy and of high quality, and that are cost effective in meeting user needs. A major component of this is system cost and operational effectiveness determination, and the development of processes that ensure that products are cost effective. This requires the integration of a number of engineering disciplines necessary for the definition, development, and deployment of complex systems. It also requires attention to the lifecycle process used to produce systems, and the integration of systems, including legacy systems, at various architectural levels.

In addition, appropriate systems management of information and knowledge across technologies, organizations, and environments is also needed to insure a sustainable world.

The journal will accept and review submissions in English from any author, in any global locality, whether or not the author is an INCOSE member. A body of international peers will review all submissions, and the reviewers will suggest potential revisions to the author, with the intent to achieve published papers that:

- Relate to the field of systems engineering;
- Represent new, previously unpublished work;
- Advance the state of knowledge of the field; and
- Conform to a high standard of scholarly presentation.

Editorial selection of works for publication will be made based on content, without regard to the stature of the authors. Selections will include a wide variety of international works, recognizing and supporting the essential breadth and universality of the field. Final selection of papers for publication, and the form of publication, shall rest with the editor.

Submission of quality papers for review is strongly encouraged. The review process is estimated to take three months, occasionally longer for hard-copy manuscripts.

Systems Engineering operates an online submission and peer review system that allows authors to submit articles online and track their progress, throughout the peer-review process, via a web interface. All papers submitted to Systems Engineering, including revisions or resubmissions of prior manuscripts, must be made through the online system. Contributions sent through regular mail on paper or emails with attachments will not be reviewed or acknowledged.

All manuscripts must be submitted online to Systems Engineering at ScholarOne Manuscripts, located at:

http://mc.manuscriptcentral.com/SYS

Full instructions and support are available on the site, and a user ID and password can be obtained on the first visit.
Inside this issue

FROM THE EDITOR-IN-CHIEF 6

SPECIAL FEATURE 7

Loss-Driven Systems Engineering (LDSE) 7
Unifying Loss-Driven Systems Engineering Activities 9
Integrating Loss-Driven Systems Engineering Activities 14
Role of LDSE for a Hypothetical Manned Space Rescue Vehicle 19
An Early Attempt at a Core, Common Set of Loss-Driven Systems Engineering Principles 22
Harmonizing the Domains of Loss-Driven Systems Engineering 27
Loss-Driven Systems Engineering and Siloism 32

BONUS ARTICLE 34

Systems Engineering the Conditions of the Possibility (Towards Systems Engineering v2.0) 34
INFORMATION ABOUT INCOSE
INCOSE’s membership extends to over 18,000 individual members and more than 100 corporations, government entities, and academic institutions. Its mission is to share, promote, and advance the best of systems engineering from across the globe for the benefit of humanity and the planet. INCOSE chapters chapters worldwide, includes a corporate advisory board, and is led by elected officers and directors. For more information, click here: The International Council on Systems Engineering (www.incose.org)

OVERVIEW
INSIGHT is the magazine of the International Council on Systems Engineering. It is published four times per year and features informative articles dedicated to advancing the state of practice in systems engineering and to close the gap with the state of the art. INSIGHT delivers practical information on current hot topics, implementations, and best practices, written in applications-driven style. There is an emphasis on practical applications, tutorials, guides, and case studies that result in successful outcomes. Explicitly identified opinion pieces, book reviews, and technology roadmapping complement articles to stimulate advancing the state of practice. INSIGHT is dedicated to advancing the INCOSE objectives of impactful products and accelerating the transformation of systems engineering to a model-based discipline.

Topics to be covered include resilient systems, model-based systems engineering, commercial-driven transformational systems engineering, natural systems, agile security, systems of systems, and cyber-physical systems across disciplines and domains of interest to the constituent groups in the systems engineering community: industry, government, and academia. Advances in practice often come from lateral connections of information dissemination across disciplines and domains. INSIGHT will track advances in the state of the art with follow-up, practically written articles to more rapidly disseminate knowledge to stimulate practice throughout the community.

ARTICLE SUBMISSION
INSIGHT@incose.org

Publication Schedule. INSIGHT is published four times per year.

Issue and article submission deadlines are as follows:
- March 2020 issue – 2 January
- June 2020 issue – 2 April
- September 2020 issue – 1 July
- December 2020 issue – 1 October

For further information on submissions and issue themes, visit the INCOSE website: www.incose.org
INSIGHT volume 23, no. 4 is sponsored by the Lockheed Martin Corporation.

CORPORATE ADVISORY BOARD — MEMBER COMPANIES

- Georgia Institute of Technology
- IBM Corporation
- Idaho National Laboratory
- ISAE - Supaero
- ISDEFE
- ISID Engineering, LTD
- ITID, Ltd
- Jacobs
- Jama Software
- Jet Propulsion Laboratory
- John Deere
- Johns Hopkins University
- KBR
- KEIO University
- L3Harris Technologies
- Leidos
- Lockheed Martin Corporation
- Los Alamos National Laboratory
- ManTech International Corporation
- Maplesoft
- Massachusetts Institute of Technology
- MBDA (UK) Ltd
- Missouri University of Science & Technology
- MITRE Corporation, The
- Mitsubishi Aircraft Corporation (Heavy Industries Group)
- National Aeronautics and Space Administration (NASA)
- National Security Agency Enterprise Systems
- Naval Postgraduate School
- Nissan Motor Co, Ltd
- No Magic Inc.
- Noblis
- Northrop Grumman Corporation
- Pennsylvania State University
- Perspecta
- Prime Solutions Group, Inc
- Project Performance International (PPI)
- Raytheon Corporation
- Roche Diagnostics
- Rolls-Royce
- Saab AB
- Safran Electronics and Defense
- SAIC
- Sandia National Laboratories
- Shell
- Siemens
- Sierra Nevada Corporation
- Singapore Institute of Technology
- Skoltech
- SPEC Innovations
- Stellar Solutions
- Stevens Institute of Technology
- Strategic Technical Services LLC
- Swedish Defence Materiel Administration (FMV)
- Systems Engineering Directorate
- Swedish Defence Materiel Administration (FMV)
- Systems Engineering Directorate
- Systems Planning and Analysis
- Thales
- TNO
- Trane Technologies
- Tsinghua University
- TUS Solution LLC
- UC San Diego
- UK MoD
- United Technologies Corporation
- University of Alabama in Huntsville
- University of Arkansas
- University of Connecticut
- University of Maryland
- University of Maryland, Baltimore County
- University of Michigan, Ann Arbor
- University of New South Wales, The, Canberra
- University of Southern California
- University of Texas at Dallas
- University of Texas at El Paso (UTEP)
- US Department of Defense
- Veoneer
- Vitech
- Volvo Construction Equipment
- Woodward Inc
- Worcester Polytechnic Institute- WPI
- Zuken Inc

CONTACT

Questions or comments concerning:
Submissions, Editorial Policy, or Publication Management
Please contact: William Miller, Editor-in-Chief
insight@incose.org

Advertising — please contact:
Susan Blessing, Senior Account Manager Sciences
Sciences, Corporate Sales
Mobile: 247- 201-723-3129
E-mail: slsbless@wiley.com

Member Services — please contact: info@incose.org

ADVERTISER INDEX

Systems Engineering Call for Papers
Annual INCOSE International Workshop
INCOSE regional events
It is our pleasure to announce the December 2020 INSIGHT issue published cooperatively with John Wiley & Sons as a systems engineering practitioners magazine. The INSIGHT mission is to provide informative articles on advancing the state of the systems engineering practice. The intent is accelerating knowledge dissemination to close the gap between the practice state and the state of the art as Systems Engineering, the Journal of INCOSE also published by Wiley, captured. INCOSE thanks corporate advisory board member Lockheed Martin for sponsoring INSIGHT in 2020 and welcomes additional sponsors, who may contact the INCOSE marketing and communications director at marcom@incose.org.

Current and future systems and systems of systems context is non-determinism with exponential increases in scale, hyperconnectivity, human influences, and thus, complexity. This INSIGHT theme is loss-driven systems engineering (LDSE) complementing capabilities-based systems engineering. Loss-driven systems engineering proactively mitigates potential losses by systemically leveraging commonalities and synergies across specialty areas such as safety, security, operational risk, resilience, critical infrastructure protection and recovery, and the ‘ilities.’ We thank theme editor John Brits and the authors for their contributions coming from INCOSE working groups, especially resilience and security, beginning in 2017. John leads with his article defining LDSE and synopsizing each themed article.

We thank author Keith Willette for also contributing his second article titled “Systems Engineering the Conditions of the Possibility (Towards Systems Engineering v2.0)” apart from his loss-driven systems engineering themed article while embracing the loss-driven theme. Keith is a proactive proponent in the systems community future of systems engineering (FuSE) initiative. He describes traditional systems engineering as focusing on cause and effect to achieve a desired outcome through human intervention. Keith states we now have the tools to transcend cause-effect and effectively embrace the nondeterministic, flexibly defined, blurred-boundaries, highly combinatorial if not infinite, and adaptability. Systems engineers can design solutions to adapt to predictable and unpredictable change for the system to remain viable while encountering adversity (loss-driven) and relevant when threatened by obsolescence (opportunity-driven).

We hope you find INSIGHT, the practitioners’ magazine for systems engineers, informative and relevant. Feedback from readers is critical to INSIGHT’s quality. We encourage letters to the editor at insight@incose.org. Please include “letter to the editor” in the subject line. INSIGHT also continues to solicit special features, standalone articles, book reviews, and op-eds. For information about INSIGHT, including upcoming issues, see https://www.incose.org/products-and-publications/periodicals#INSIGHT.
Theme Editor’s Introduction

Loss-Driven Systems Engineering (LDSE)

John S. Brits, jbrits@johnsbrits.com
Copyright ©2020 The MITRE Corporation. All rights reserved. Approved for public release. Distribution unlimited 20-02802-1

In 2017, INCOSE working groups (especially resilience and security) investigated their potential commonalities and made attempts to clarify how they overlapped and how they differed. This led to realizing these specialty areas had commonalities and synergies, needing exploration to benefit many systems engineering specialty areas and systems engineering overall. The term “loss-driven systems engineering” identifies this common interest area. INCOSE International Workshop 2018 held an exploratory meeting on loss-driven systems engineering. There, participants agreed this concept needed pursuing and decided, as a first step, to pursue an INSIGHT, an INCOSE magazine, special theme issue on loss-driven systems engineering. This issue is the result.

While much of systems engineering focuses on the system delivering desired capabilities, loss-driven systems engineering specialty areas have a different aim: they address the potential losses associated with developing or using systems. Numerous specialty areas such as safety, security, operational risk, resilience, critical infrastructure protection and recovery, and numerous so called ‘ilities’ address loss-driven systems engineering. In this issue we explore loss-driven systems engineering’s meaning and the way it can bring value to systems engineering.

LOSS-DRIVEN SYSTEMS ENGINEERING

Definition: loss-driven systems engineering is the value adding unification of the systems engineering specialty areas that address potential losses associated with systems. Example loss-driven specialty areas include: resilience, safety, security, operational risk, environmental protection, quality, and availability.

- There is commonality and synergy among these specialty areas, which systems engineering should address.

- These specialty areas have potential synergies
  - Shared loss scenarios
  - Shared requirements
  - Shared modeling and analysis techniques
  - Shared architecture and design solutions
  - Shared risk management

- More explicitly addressing and unifying LDSE offers significant benefits
  - Reducing engineering effort by eliminating redundant activities among the specialty areas
  - Comprehensive consideration of possible losses
  - Effective solutions addressing multiple loss-driven specialty area interests
  - Eliminating conflicts among the loss-driven solutions
  - Reducing the data load generated by multiple specialty areas to a minimal, non-redundant set
  - Mutual learning among the loss-driven specialty areas

“Unifying Loss-Driven Systems Engineering Activities” by John Brits and Michael McEvilley, explores the loss-driven systems engineering meaning, identifies the existing systems engineering specialty areas falling under the loss-driven systems engineering umbrella, and discusses the potential for unifying those specialty areas along their shared characteristics. The article proposes the characteristics which make unification possible: asset types, loss types, adversity types, requirements, and architecture and design solutions. It considers the benefits expected from unifying loss-driven specialty areas, discusses how LDSE might be implemented, and explores the effect such a unification will have on systems engineering throughout the system life cycle.

"Integrating LDSE Activities" by David Endler considers real life integration of loss-driven systems engineering activities into system development activities. Challenges identified include, lack of appreciation of the importance of loss-driven systems engineering activities and organizational barriers. The article proposes methods to overcome those barriers based on widely accepted standards. The author finds that existing systems engineering standards poorly describe loss-driven systems engineering activities and fail to integrate loss-driven activities with traditional engineering activities. The paper provides an approach to successfully accomplish the needed integration with emphasis on the need that loss-driven systems engineers participate throughout the life cycle and be supported by a common understanding of an integrated approach.

"Role of LDSE for a Hypothetical Manned Space Rescue Vehicle" by Ken Cureton examines the utility of Loss-Driven Systems Engineering via a thought experiment regarding desirable characteristics for achieving resilience, safety, reliability, security, and other loss-driven goals. Various design reference missions explore assessing required loss-driven capabilities in automated flight operations for a hypothetical Manned Space Rescue Vehicle. Central to this assessment is identifying key adversities to achieving mission success and evaluating methods to avoid, withstand, and recover from the loss such adversities cause. Such methods apply classical technical disciplines such as resilience, safety, reliability, survivability, and security in an integrated fashion, while recognizing each discipline’s expertise, proven methods, tools, and techniques. This article also examines the importance of flexible and creative crew actions and adaptive systems for overcoming unexpected adversity.
“An Early Attempt at a Core, Common Set of Loss-Driven Systems Engineering Principles” by Mark Winstead investigates truths that apply throughout the loss-driven systems engineering discipline, and thus guide its application. The article looks for commonality and similarities among principles previously articulated for loss-driven specialty areas (safety, security, resilience, and critical information protection and recovery). The author then pursues more fundamental, and abstract principles unified across specialties. Mark identifies “new” transcendent principles and presents a core set of principles for loss-driven systems engineering.

In “Harmonizing the Domains of Loss-Driven Systems Engineering” Keith Willett proposes a systems engineering framework for considering loss-driven systems engineering. He considers a system’s characteristics, which include what it is (structure, state), what it does (function, behavior), where it resides (environment, containing whole), what it uses (resources, energy source, raw material), what it contains (content), and why it exists (value delivery). He characterizes adversity as a disturbance inducing stress on a system causing loss in one or more characteristic.

Finally, in “LDSE and Siloism,” Scott Jackson discusses the siloism concept relating to LDSE and ways to mitigate its effects. Siloism is any team member’s unwillingness to share information. Failure to mitigate siloism can reduce team effectiveness. A recognized siloism mitigation method is the integrated product team (IPT). The IPT uses organizational structure and rigorous management to encourage sharing of information among specialties. By aligning the organizational structure of the project to the physical architecture of the system, close cooperation among specialties is facilitated. IPTs are part of the larger concept called Integrated Product and Process Development (IPPD).

SUMMARY

Loss-driven systems engineering offers a valuable framing for numerous existing loss-driven systems engineering specialty areas. Adding and using this framing in the systems engineering communities can integrate many currently isolated systems engineering activities, promising improved system effectiveness, and reduced systems engineering costs—while improving the management of potential losses associated with developing and using systems.

ABOUT THE AUTHOR

John S. Brits is a systems engineer working for the MITRE Corporation. John has degrees in physics engineering, nuclear engineering, and systems engineering. He has worked in the nuclear power industry where he specialized in radiation protection, the IT industry where he focused on AI applications to complex engineering decision-making, and the systems engineering consulting arena where he has supported aerospace activities, focusing on resilience. John is a past INCOSE Resilient Systems Working Group chair and currently leads the INCOSE Loss-Driven Systems Engineering Initiative. John is a registered professional engineer, a project management professional, and a certified systems engineering professional.
Unifying Loss-Driven Systems Engineering Activities

John Brtis, jbrtis@ohnsbrtis.com; and Michael McEvilley, mcevilley@mitre.org
Copyright © 2020 The MITRE Corporation. All rights reserved. Approved for public release. Distribution unlimited 20-02802-2

ABSTRACT
Systems, by definition, deliver desired capability. Not surprisingly—much of systems engineering is capability-driven; it focuses on developing systems to deliver desired capability. There are, however, non-capability-driven areas of systems engineering. Loss-driven systems engineering is an example. Loss-driven systems engineering concerns itself with addressing the possible losses associated with system development and use. This paper explores loss-driven systems engineering’s meaning, identify the existing systems engineering specialty areas under the loss-driven systems engineering umbrella, and discuss the potential for unifying those specialty areas along the attributes they share. We believe the attributes for possible unification include: asset types, loss types, adversity types, requirements, and architecture and design solutions. We identify the likely benefits expected from such a unification, and we explore the effect such a unification will have on systems engineering throughout the system life cycle.

KEYWORDS: systems engineering, loss-driven, capability, adversity, modeling, resilience, safety, security.

INTRODUCTION
A system is an arrangement of elements that together exhibit capability that the individual elements do not. That capability is the system’s purpose, and the reason we value the system. Unsurprisingly, systems engineering methodologies focus on capability delivery. Methodology sources such as the Systems Engineering Handbook (Walden et al. 2015), the Systems Engineering Body of Knowledge (SEBOK 2019) and ISO/IEC/IEEE 15288 (ISO 2015) provide full life cycle and fully integrated methodologies focusing on generating and deploying systems to deliver capabilities. Those methodologies are largely capability-driven. Loss and loss-driven specialty areas are largely treated in isolation.

Loss-driven systems engineering addresses the possible losses associated with system development, use, and sustainment. Examples of loss-driven concerns include reliability, availability, maintainability, safety, security, survivability, risk, and resilience. The systems engineering community has developed numerous systems engineering specialty areas that address these concerns in various ways.

Loss-driven systems engineering specialty areas, often treated separately, do not often fully integrate into the overall systems engineering methodology. The authors conclude there is a commonality among these specialty areas not yet exploited, which can improve systems engineering efficiency and effectiveness. In the past, commonality among the loss-driven specialty areas has sometimes been recognized; as with reliability, availability, and maintainability, which often combine under the acronym “RAM” (Walden et al. 2015). We believe the commonality and potential synergy among the broader range of loss-driven specialty areas to be significant and believe their unification can achieve increased engineering efficiency and effectiveness.

THE BASIS OF COMMONALITY
In the 2017 to 2019 timeframe, the authors explored the commonality of “protecting against loss” in security and safety in their MITRE work. In parallel, the INCOSE resilience and security working groups looked into how their specialty areas relate to one another. The authors led those activities.

There was a spectrum of beliefs in the resilience, safety, and security communities regarding definitions, scope, and the objectives and means of achieving safety, resilience, and security. This range of beliefs made precisely identifying the overlaps and distinctions of these domains intractable.

Fortunately, a precise understanding of these issues was not necessary. Systems engineering seeks to address the system as a whole. Systems engineering must address all specialty areas in a mutually supportive and optimized manner, and must do so independent of the specialties’ unique definitions, scopes, objectives, and means. What matters is meeting all resilience, safety, and security objectives. Thus, the exact demarcation between the different specialty areas becomes unimportant.

Further resilience, safety, and security often have common objectives, concepts and principles, requirements, architectural solutions, design solutions, analyses, and
methodologies. In addition, it became clear the engineers individually responsible for resilience, safety, or security can often do a better job if they work collaboratively with one another. This is because they all have the same overarching concern: addressing possible losses associated with the system of interest.

Given the commonality among these three loss-driven specialty areas, inspecting the Systems Engineering Handbook (Walden et al. 2015) identified other specialty engineering areas sharing loss-driven systems engineering concerns. Those identified include:

- availability
- environmental impact
- maintainability
- resilience engineering
- reliability
- risk management
- system safety engineering
- system security engineering
- quality management
- a number of recognized -ilities.

This paper will explore the potential benefit of integrating the loss-driven systems engineering specialty areas: their vocabularies, their objectives, their methodologies, and how they achieve their ends.

**LOSS-DRIVEN ATTRIBUTES SHARED BY THE LOSS-DRIVEN SPECIALTY AREAS AND THE POTENTIAL FOR UNIFICATION**

The loss-driven specialty areas share attributes specifying the scope of each specialty area. All loss-driven specialty areas have as attributes:

- the asset types considered
- the loss types addressed
- the adversity types addressed
- the coping strategies considered
- the system aspects and its environment under consideration

The values of these attributes establish the scope of each specialty area. These scopes can differ among the loss-driven specialty areas, but in many cases, they overlap, or are identical. These loss-driven attributes and the possibility of aggregating their overall scopes, provide a basis for integrating the loss-driven specialty areas, by aggregating the parameter value range and then addressing their aggregate scopes. We consider each attribute below.

**ASSET TYPES CONSIDERED**

An asset is something of value. The asset types considered in the various loss-driven specialty areas includes such disparate tangible and intangible items as:

- abilities, capabilities, functionality, services
- advantage (competitive, combatant, technical)
- data and information
- environment (air, land, water, space, cyberspace)
- equipment, facilities, structures, infrastructure
- hardware, software, firmware
- human life and health
- image, reputation, trust
- intellectual property
- money, property, investment
- processes, procedures
- system elements, systems, systems of systems

Different loss-driven specialty areas focus on particular asset types. For example, safety tends to focus on human life and health, and environmental assets. Resilience and availability focus on the system of interest capability, while security considers any asset of value.

Loss-driven systems engineering must address the full range of asset types.

**LOSS TYPES CONSIDERED**

A loss is any reduction of the ability to satisfy the stakeholder desires and needs. Once we identify an asset of interest, we can determine the possible loss types. The particular asset loss type may be one or more of:

- outright destruction
- integrity loss (damage, modification)
- capability, function, adaptability, compatibility loss (total, degraded)
- availability, accessibility, usability loss (total, diminished)
- control loss (total, diminished)
- advantage loss (combatant, technical, competitive)
- ownership, possession loss (copied, forged, stolen)
- quality loss (performance, correctness, reliability, accuracy, precision, satisfaction)
- image, reputation, trust loss
- loss in terms of sensitivity (confidentiality, privacy)

Loss-driven systems engineering must address the full range of loss types.

**ADVERSITY TYPES CONSIDERED**

An adversity is anything that can contribute to a loss. The adversity types considered are another important characteristic varying among the loss-driven specialty areas. Examples of the types of possible adversities include:

- Natural | human | machine
- Friendly | neutral | opponent
- Intentional | unintentional
- Malicious | non-malicious
- Threats | attacks
- Misuse | abuse
- Chronic | acute | intermittent
- Known | unknown
- Nominal | abnormal
- Hazards | vulnerability
- Defects | exposure | flaws | weaknesses
- Machine faults | errors | failures
- Emergence | side effects
- Human errors of omission | commission
- From within the system | from the system's environment
- Scarcity of personnel | material | consumables
- Enabling system and resource degradation

Loss-driven specialty areas vary in the adversity types falling within their scope. Some consider only external threats. Some only consider malicious, intentional, human threats. Others consider non-human, non-malicious, and non-intentional adversities.

Loss-driven systems engineering must address the full range of adversity types.

**COPING TECHNIQUE TYPES CONSIDERED**

Coping techniques are architecture, design, and operational choices that address possible loss. Coping techniques can apply at the component, subsystem, system, or the system of systems level. Coping techniques can include physical elements, human elements, and conceptual elements.

Loss-driven systems engineering fundamental objectives include the following:

- Prevent the occurrence of loss
- Limit the extent of loss
- Recover from loss

The scope and types of coping strategies for achieving these objectives vary among the various loss-driven systems engineering specialty areas. Coping strategy examples include:

- Eliminate or reduce the source of the causal event or condition (the adversity)
- Eliminate or reduce likelihood of loss
- Risk, issue, and opportunity management
- Contingency management
- Avoid | withstand | recover | evolve | accept
- Absorption, adaptation, agility, anticipation, preparation, prevention, constraining, functional redundancy, layered defense physical redundancy, redeploying, disaggregation, monitoring

Loss-driven systems engineering must address the full range of coping techniques.
**SYSTEM AND ENVIRONMENT ASPECTS AND TYPES CONSIDERED**

System and environment aspects vary among the loss-driven systems engineering domains. Some consider only particular system segments (information and information technology-focused loss management). Some consider only certain system types (critical infrastructure); some consider only the system of interest, but not the systems of systems or the component level.

Loss-driven systems engineering must address the full range of systems aspects, enabling systems, supporting systems, and environmental aspects, system types, and system aggregation levels.

**SYSTEMS ENGINEERING CONSEQUENCES**

Loss-driven considerations of the system of interest should be addressed holistically. Early in the life cycle, when identifying the conditions under which the system must operate and the key system characteristics, the systems engineer must also identify the assets of interest and the possible loss types (consistent with the stakeholder desires and priorities). Such information should be identified in a unified manner, and should be sufficient to address the relevant loss-driven systems engineering specialty areas. As the life cycle proceeds, holistically developed loss-driven requirements should address the full aggregate scope of loss-driven specialty areas. The same consideration should be applied to trades among the various architectural and design solutions.

While a holistic and unified approach among the various loss-driven systems engineering domains is critical, we believe it is also important not to lose the unique approaches developed in any individual loss-driven specialty area. Each specialty area may bring a unique—and valuable—approach to achieving its particular ends. Including the wisdom of the various loss-driven domain should be ensured.

**SYSTEMS ENGINEERING LIFE CYCLE PROCESS CONSEQUENCES**

Brits and McEvilley (2019) assessed the modifications needed to adequately address resilience (a loss-driven specialty area) in two standard systems engineering life cycle processes, the INCOSE Systems Engineering Handbook (Walden et al. 2015) and ISO/IEC/IEEE 15288:2015(E) (ISO 2015). Most needed augmentations occurred early in the life cycle. Processes requiring augmentations are:

- Business or mission analysis process
- Stakeholder needs and requirements definition process
- System requirements definition process
- Architecture definition process
- Design definition process
- Risk management process
- Business or Mission Analysis Process
- Defining the problem space should include identifying adversities under which the system must provide capability and the expectations for possible loss types and acceptable loss under those adversities.
- The operations concept and solution classes characterizing the solution space
should consider the system’s ability to deal with adversities for the purpose of providing the required loss management capabilities.

- Evaluating alternative solution classes must consider the system’s ability to manage loss under the adversities.

**Stakeholder Needs and Requirements Definition Process**

- The stakeholder set should include persons who understand the potential adversities and the requisite stakeholder needs for loss management.
- Identifying stakeholder needs should identify stakeholder expectations for managing loss under adverse conditions and should consider degraded—but useful—operation modes.
- The operational concept should consider adversities as part of the defined operational environment. The scenarios should include loss-driven scenarios.
- Transforming stakeholder needs into stakeholder requirements should include developing stakeholder loss management requirements.
- Analyzing stakeholder requirements should include appropriate adversity scenarios, including the intended operational environment.

**System Requirements Definition Process**

- Identifying quality requirements should consider loss management.
- System requirements that manage loss will often address system -ilities. Achieving loss management and the -ilities should be addressed holistically.

**Architecture Definition Process**

- The selected architecture viewpoints should support loss management representation.
- Experience shows loss management requirements can significantly limit the range of acceptable architectures. Loss management requirements must be fully mature, and fully validated and verified when used for architecture selection.
- Individuals developing candidate architectures should be familiar with architectural techniques for achieving loss management.
- Architectural techniques for achieving loss management are often germane to the system -ilities. Achieving loss management and the -ilities should be addressed holistically.

**Design Definition Process**

- Individuals developing candidate designs should be familiar with design techniques for achieving loss management.
- Design techniques for achieving loss management are often germane to the system -ilities. Achieving resilience and the -ilities should be addressed holistically.

**Risk Management Process**

It is important to recognize that loss management and risk are tightly coupled. Risk management activities should explicitly plan for coordination with loss management activities.

**MODEL-BASED SYSTEMS ENGINEERING CONSEQUENCES**

Model-based systems engineering data and models need to be augmented to address the shared loss-driven systems engineering attributes: assets, losses, adversities, and coping techniques and the common information artifacts identified above. Table 1 identifies some additional modeling information for capture during the various life cycle stages to support effective development and documentation of loss management scenarios and loss management requirements. For this discussion we assume the Systems Modeling Language (SysML) is the language used. It is worth noting some loss-driven information requirements have a more complex content and structure than capability-driven information and requires formal patterns.

**POTENTIAL LOSS-DRIVEN VIEWPOINT BENEFITS**

We expect numerous advantages and benefits from incorporating the unified, loss-driven viewpoint into the systems engineering processes.

- Reducing engineering effort by eliminating redundant efforts among the specialty areas
- Helping to ensure a comprehensive consideration of losses
- Ensuring coherence and elimination of conflicts among the loss-driven solutions
- Identifying highly effective solutions addressing multiple loss-driven specialty area interests
- Providing a holistic viewpoint addressing the multiple loss-driven perspectives
- Reducing the data load generated by multiple specialty areas to a minimal, non-redundant set
- Mutual learning among the loss-driven specialty areas

**RECOMMENDATIONS**

To address all loss-driven specialty areas the systems engineer should holistically consider:

- the full adversity spectrum
- the full weakness, defect, flaw, exposure, hazard, and vulnerability spectrum
- the full asset and loss spectrum
- the full timeframe of interest spectrum
- the full coping mechanism spectrum
Further, the systems engineer should:
- elicit, analyze, and capture loss-driven requirements as part of the overall stakeholder and system requirements development.
- make the loss-driven architectural decisions holistically across the loss-driven specialty areas
- make the loss-driven design decisions holistically across the loss-driven specialty areas
- integrate the management of risks associated with all loss-driven areas into the project’s risk management

All considerations should be based on the stakeholder desires, needs, and priorities.

In the event such a comprehensive approach is not possible, we suggest as a minimum the various engineers in the loss-driven specialty areas establish a means of working in concert.

SUMMARY
We have found a significant commonality and potential for synergy among the loss-driven systems engineering specialty areas. We have found the loss-driven systems engineering specialty areas share numerous key attributes and associated property values both overlap and differ.

Aggregating the values of these properties can bring the individual specialty areas together in an integrated framework for loss-driven systems engineering. We have identified several reasons to believe an integrated approach to loss-driven systems engineering will improve systems engineering effectiveness in addressing loss-driven issues. Finally, we identified the need to extend the systems engineering life cycle and model-based systems engineering models to address loss-driven systems engineering.

REFERENCES

ABOUT THE AUTHORS
John S. Brits is a systems engineer working for the MITRE Corporation. John has degrees in physics engineering, nuclear engineering, and systems engineering. He has worked in the nuclear power industry where he specialized in radiation protection, the IT industry where he focused on AI applications to complex engineering decision-making, and the systems engineering consulting arena where he has supported aerospace activities, focusing on resilience. John is a past INCOSE Resilient Systems Working Group chair and currently leads the INCOSE Loss-Driven Systems Engineering Initiative. John is a registered professional engineer, a project management professional, and a certified systems engineering professional.

Michael A. McEvilley is a principal scientist in the Systems Engineering Technical Center of The MITRE Corporation. He supports the Department of Defense (DoD) in improving systems engineering assured effectiveness to deliver weapon systems suitable for operation in contested cyberspace. He has extensive experience in high confidence software-intensive systems, requirements engineering, and secure operating system design and evaluation. Michael has a Master of Science degree in computer science from The George Washington University.
Integrating Loss-Driven Systems Engineering Activities

David Endler, de@davidendler.de
Copyright ©2020 by David Endler. Published and used by INCOSE with permission.

ABSTRACT
Loss-driven systems engineering activities are key to realizing successful systems. At the same time, loss-driven systems engineering assessments are, in most cases, complex. In real life projects, integrating loss-driven systems engineering activities in the system development activities might be difficult. In some cases, there is a lack of understanding the activities’ importance and sometimes there are organizational barriers. To overcome those barriers, we propose an approach based on widely accepted standards. The difficulty is most existing systems engineering standards poorly describe loss-driven systems engineering activities and how they integrate with traditional engineering activities. This paper provides an approach to successfully accomplish this integration. It is extremely important to involve loss-driven systems engineers in every life cycle phase. At the same time, achieving a common integrated approach understanding is necessary.

KEYWORDS: Loss-Driven Systems Engineering; Integration; Development Process; Reliability; System Safety; Availability; Maintainability; Security; Resilience

INTRODUCTION
Developing complex systems involves various stakeholders with conflicting interests. Typically, a project manager, who delegates technical aspect responsibility to a lead systems engineer, leads these projects. Consequently, the lead systems engineer carries the responsibility to establish trade studies balancing conflicting technical needs and requirements to realize a successful system. Establishing integrated teams, comprising specialists from many different domains, achieves this. The INCOSE Fellows “systems engineering” definition reflects this very well, stating “systems engineering is a transdisciplinary and integrative approach (INCOSE 2020).”

While integrated team members may include many more experts (purchasing, marketing), this paper addresses the relationship between traditional engineering activities (mechanical engineering, electrical engineering) and loss-driven systems engineering (LDSE) activities (reliability, availability, maintainability, safety (RAMS), and security, resilience, and recovery). Figure 1 shows an integrated team example with members from different engineering disciplines.
In an integrated team, the lead systems engineer is responsible for identifying agreeing interfaces, defining and allocating system requirements to the corresponding system elements, resolving conflicting requirement issues, and many more activities. In particular, they also must ensure respective specialists perform all analysis tasks required to develop functional and physical system architectures.

Many projects observe subliminal conflicts between traditional engineering and LDSE disciplines due to the very different LDSE discipline natures: where traditional engineering focuses on required capability delivery, LDSE addresses potential system of interest associated losses. Typical examples include, on one hand, obviously contradicting cost and functionality requirements and, on the other hand, safety and reliability requirements. Also observed, traditional field engineers directly allocating to one project. In many cases, loss-driven systems engineers allocated to several (sub-)systems or even to several different projects within the same organization. Consequently, integrated team affiliation was much stronger for traditional engineers compared to loss-driven systems engineers.

There are numerous examples of systems having poor RAMS properties such as Australia’s Collins Class submarines (Defense Industry Daily 2015).

This paper proposes an approach to integrating all engineering disciplines to develop a system optimized for all disciplines involved (cost, functionality, reliability, and safety).

**PROCESS DESCRIPTIONS**

The first step to resolve conflicts between the parties involves analyzing process descriptions, identifying how LDSE aspects integrate into the system development.

**Systems Engineering Process Descriptions**

Typically, engineers from the traditional domain can easily apply systems engineering process descriptions such as ISO 15288:2015. The ISO 15288:2015 technical process activity description identifies LDSE aspects (clause 6.4.2.3 d(2) on page 53 or clause 6.4.3.3 b(3) on page 55). References include other ISO standards such as IEC 61508 (Functional safety) or ISO TR 18529 (Ergonomics), while not covering aspects like reliability, availability, or maintainability.

Interviews with engineers involved in traditional engineering show understanding process descriptions help their domain. So they focus on system functions and performance requirements, easily overlooking LDSE aspects. ISO 15288 appendix E.4 describing specialty engineering views amplifies this.

The same is true for other standard works such as BKCASE Systems Engineering Body of Knowledge (SEBoK) or INCOSE Systems Engineering Handbook Version 4. SEBoK Part 3 “Systems Engineering and Management” addresses requirements and logical architectures. Part 6 “Related Disciplines” covers LDSE aspects. Looking closer into INCOSE Systems Engineering Handbook Version 4 the picture is the same: details about the technical ISO15288 processes found in chapter 4 whereas chapter 10 describes LDSE activities (and others).

People become even more confused when trying to understand graphical LDSE activity representations in SEBoK and
INCOSE Systems Engineering Handbook Version 4. Figure 2 shows the integration process for specialty engineering activities from SEBoK v2.1.

The SEBoK figure shows many details and for some arrows the reader must figure out what they represent. Figure 3, taken from INCOSE Systems Engineering Handbook Version 4, carries even more details, more arrows, and raises many questions. The reader starts at the figure’s upper right corner, then reads the questions on left upper corner, and finally gets lost in the arrow wilderness. Without studying the reference given it is impossible to understand the so-called “fundamental equation of sustainment” in INCOSE’s work (2015).

In summary, current practice treats LDSE activities separately from traditional engineering activities and the LDSE activity presentation does not contribute to an integrated approach. LDSE activity importance receives very poor awareness. Consequently, this may lead to situations considering LDSE aspects too late in the development.

**Loss-Driven Systems Engineering Process Description**

The pictures changes when assessing LDSE activity standards. Standards from this domain do not only emphasize integrating LDSE activities is important in product development processes, it also describes how to accomplish this. As an example, Figure 4 shows a safety assessment process model for aircraft system safety assessments taken from SAE ARP 4754A. (see Figure 4 on the next page.)

Just like in process descriptions presented above, standards related to other LDSE activities provide a systems engineering activity overview at the very beginning (MIL-HDBK-338B Electronic Reliability Design Handbook chapter 4.2).

It strongly emphasizes LDSE activities heavily rely on the results from the traditional domains. At the same time, it explains LDSE activities make an essential contribution to successfully realizing the system, providing answers to questions like “How do we know when the design is adequate?” or “How is the effectiveness of a system measured?” (DoD 1998) when applying the methods defined.

Unfortunately, the approach presented limits validity to the LDSE activity area under consideration. Therefore, the approach cannot include other areas as well.

**INTEGRATED APPROACH**

The approach presented bases itself on experience gathered in several different industry projects. Even though the difficulties in integrating LDSE activities varied in severity, the fundamental root cause was always lack of understanding. Achieving a common LDSE importance and integration understanding requires an approach considering both domains.

**Mutual Appreciation**

Practice proves a process description close to the traditional process descriptions like ISO 15288 works to create mutual appreciation on both sides, traditional and LDSE.

The approach used bases on a systems engineering process description described in ISO 26702:2007 Systems engineering—systems engineering process application
and management. Typically, both domains accept this standard even though it has not updated for quite some time. This standard starts with describing an integrated approach (ISO 26702:2007 chapter 1.1) and maintains this approach throughout the document (chapter 4.7.4, Table-1, or chapter 6.1.1). However, a document with more than 100 pages is not well-suited to gain engineer interest. Therefore, the aim to have something simple in hand helping achieve common understanding continues.

Finally, Figure 5 (on the next page) provides a simple approach. This figure, taken from ISO 26702:2007, has proven its worth in practice. Modifying the boxes on the figure’s right-hand side created common understanding. Those boxes adapt depending on the subject under consideration. The assessments performed do not only cover traditional aspects but LDSE activities as well.

Figure 5’s big advantage is it comes from a widely accepted standard in the traditional domain. Also, it is very easy to show this is exactly the way to use the figure, referring to ISO 26702:2007 chapter 6.7.6. This chapter explicitly states each trade study life cycle must consider cost (chapter 6.7.6.1) and system safety aspects (chapter 6.7.6.3).

**Embedding**

The approach shown so far will create a common understanding. However, this might not be enough to anchor this common understanding in a sustainable way. To achieve this, we propose a workshop led by the lead systems engineer to establish a Figure 5 tailored version reflecting the current project’s particular situation.

In this workshop, respective domains agree on the created results and place the arrows going from left to right and back. This reveals required rigor levels for assessments from the traditional domain feeding into the assessments performed by loss-driven systems engineers. It will also show the ways the assessments performed by loss-driven systems engineers influence the system design. The lead systems engineer will guarantee the agreements made align with any other project constraints.

This process interface definition—a Figure 5 tailored version—must document under configuration control. It has been very effective to print this tailored version as a large-size poster to put in the corresponding offices.

Figure 4. Safety Assessment Process Model, see SAE (2010) Figure 7

**REFERENCES**

Figure 5. Systems Engineering Process, see ISO 2007 Figure 4


ABOUT THE AUTHOR

Dr. David Endler works as a systems engineering consultant and training provider. He has participated in many large-scale projects such as systems engineering process definition for defense and renewable energies companies, lead systems engineer for major aircraft systems, safety and certification responsible for air traffic management systems. He has experience from many industries such as aerospace, automotive, renewable energies, and marine systems. Dr. Endler holds a PhD in physics from the University of Hamburg. He is the current INCOSE Technical Director (2019 to 2021), holds the SE-ZERT® Level A and INCOSE CSEP certificates and is an accredited training provider for SE-ZERT® trainings.
Role of LDSE for a Hypothetical Manned Space Rescue Vehicle

Kenneth L. Cureton, cureton@usc.edu
Copyright ©2020 by Kenneth L. Cureton. Published and used by INCOSE with permission.

ABSTRACT

This article examines Loss-Driven Systems Engineering (LDSE) utility via a thought experiment regarding harmonizing desirable characteristics for resilience, safety, reliability, security, and other loss-driven specialty areas. Various design reference missions explore and assess required loss-driven capabilities in automated flight operations for a hypothetical Manned Space Rescue Vehicle. Identifying key adversities to achieving mission success, and evaluating potential methods to avoid, withstand, and recover from loss caused by such adversities are central to this assessment. Such methods apply classical technical disciplines such as resilience, safety, reliability, survivability, and security in an integrated fashion, recognizing and respecting each discipline's expertise and proven methods, tools, and techniques. This article also examines the concurrent need to consider loss-driven solution unintended consequences: the “First Do No Harm” medical concept ensuring adversity’s “cure cannot be worse than the disease.” Finally, this article examines the potential to leverage flexibility and creativity in crew actions and adaptive systems to overcome unexpected adversity.

KEYWORDS: Loss-Drive Systems Engineering (LDSE); Resilience; Adversities; Reliability; Safety; First Do No Harm; Cyber-Physical Human Systems (CPHS); Design Reference Missions

INTRODUCTION

Systems engineering practitioners may ask, “What is the difference between Loss-Driven Systems Engineering (LDSE) and well-structured Systems Engineering?” This article presents a thought experiment regarding desirable characteristics in achieving loss-driven advantages for a hypothetical manned space rescue vehicle demonstrating LDSE utility within systems engineering. This article assesses operational architectures for several design reference missions and resulting automated flight operation (with manned control options) for required loss-driven capabilities and driving requirements. This assessment focuses on identifying key adversities to achieving mission success, and evaluating potential methods to avoid, withstand, and recover from loss caused by such adversities. Such methods apply classical technical disciplines such as resilience, safety, reliability, survivability, security, and risk/issue/opportunity management in an integrated fashion, recognizing and respecting each discipline’s expertise and proven methods, tools, and techniques. This harmonized technical discipline integration to achieve resilience method strengths—with resulting consequence mindfulness—is essentially LDSE. Detailed methods for accomplishing such harmonization surpass this article’s scope, although it provides a few examples.

DESIGN REFERENCE MISSION 1 (DRM-1): ASSURED CREW ROTATION BETWEEN EARTH AND A SPACE STATION (Daniher and Cureton 1992) (Fraser 1990)

This mission transports crew from the Earth to a generic space station via a manned vehicle, and after some time for crew operations at the space station (typically 90 days), returns the crew to Earth. The transport vehicle remains attached to the space station (in a quiescent mode but ready-for-use) until needed. For a station with permanent manned capability, the prior crew returns home on the prior transport vehicle after successful replacement crew delivery by a second transport vehicle which remains attached to the station. (This mission is like the current crew rotation at the International Space Station). Table 1 on the next page shows this mission’s driving LDSE characteristics:

Each mission activity listed must happen safely and reliably yet mindful of the major adversities and potential harms, which enormously impacts not only safety engineering and reliability engineering but also many other traditional and specialty engineering disciplines. For example, achieving required operational availability (A0) despite equipment or environmental adversity likely requires automated vehicle operation with crew involvement (ranging from initiating required automation to full manual control or override). This requires careful computer hardware and software engineering harmonization with many other vehicle engineering disciplines such as environmental control & life support systems, crew displays and controls, electrical power, communications & tracking, propulsion, structural engineering, aerosurface, guid-
### Table 1: DRM-1 Driving LDSE Characteristics

<table>
<thead>
<tr>
<th>Mission Activity</th>
<th>Major Adversities</th>
<th>Major Potential Harms</th>
</tr>
</thead>
<tbody>
<tr>
<td>Support vehicle transportation to launch facility</td>
<td>Transport</td>
<td>Transportation Personnel, Vehicles, Facilities, Environment</td>
</tr>
<tr>
<td>Support transport vehicle integration with a launch vehicle</td>
<td>Integration</td>
<td>Ground personnel, Vehicles, Facilities, Environment</td>
</tr>
<tr>
<td>Support crew ingress into the transport vehicle</td>
<td>Damage</td>
<td>Crew, Transport Vehicle, Launch Vehicle</td>
</tr>
<tr>
<td>Support the crew from the Earth to the station</td>
<td>Performance, Atmosphere, Space Environment</td>
<td>Crew, Environment, Other Vehicles (especially Space Station)</td>
</tr>
<tr>
<td>Actively dock with the station or passively berth at the station</td>
<td>Clearances, Damage</td>
<td>Crew, Transport Vehicle, Space Station</td>
</tr>
<tr>
<td>Support crew egress from the transport vehicle</td>
<td>Damage</td>
<td>Crew, Transport Vehicle, Space Station</td>
</tr>
<tr>
<td>Support safe quiescent mode once attached, monitors health and status to alert the station of any vehicle hazards</td>
<td>Atmospheric Leakage, Power System Thermal Overload, Vehicle Inability to Support Crew Return</td>
<td>Space Station, Transport Vehicle</td>
</tr>
<tr>
<td>Support crew operations refresher training and 'lifeboat drills'</td>
<td>Damage, Crew Skills</td>
<td>Crew, Transport Vehicle, Space Station</td>
</tr>
<tr>
<td>Reactivate the transport vehicle and support crew ingress from the station into the transport vehicle</td>
<td>Damage</td>
<td>Crew, Transport Vehicle, Space Station</td>
</tr>
<tr>
<td>Detach from the station and maneuver away from the station</td>
<td>Clearances, Damage</td>
<td>Crew, Transport Vehicle, Space Station</td>
</tr>
<tr>
<td>Maneuver to the necessary reentry orbital position</td>
<td>Performance, Space Environment</td>
<td>Crew, Other Space Vehicles (especially Space Station)</td>
</tr>
<tr>
<td>Support the crew during entry, descent, and landing</td>
<td>Performance, Space Environment</td>
<td>Crew, Other Vehicles (Space, Air, Sea, Ground), Environment (especially at Landing Area)</td>
</tr>
<tr>
<td>Transport the crew to the assigned landing area (on ground or at sea)</td>
<td>Transport</td>
<td>Crew, Transportation personnel, Vehicles, Facilities, Environment</td>
</tr>
<tr>
<td>Support crew egress and vehicle 'safing' after landing</td>
<td>Damage, Hazardous Materials</td>
<td>Crew, Ground/Sea Personnel, Environment</td>
</tr>
<tr>
<td>Support vehicle transportation to a processing facility for subsequent disposal or refurbishment</td>
<td>Transport</td>
<td>Transportation Personnel, Vehicles, Facilities, Environment</td>
</tr>
</tbody>
</table>

Adversity, navigation, flight control, thermal protection, structural, and mechanical systems. Mission design and operation requires security engineering involvement to mitigate malicious physical or cyber attack or operator intent (including human error). Survivability Engineering involvement is necessary to withstand variable and typically hostile space, atmospheric, and Earth surface environmental conditions.

Cyber-Physical Human Systems (CPHS) techniques are essential for this mission because some circumstances require full automation, and others may require crew override or even full manual control, possibly even on the same mission! The inherent complexity resulting from CPHS engineering is usually “worth the cost” because of risk, safety-of-life, and other loss considerations. However, all engineering efforts need to strive for elegant, coordinated solutions to reduce unnecessary complexity (Madni and Sievers 2018) (Sowe et al. 2016).

These engineering efforts must coordinate and integrate (without succumbing to sub-optimization by each discipline) to consider unintended consequences (or conflicts) of their varied required resilience achievement methods. LDSE draws upon the “First Do No Harm” medical concept to ensure adversity’s “cure cannot be worse than the disease.” Therefore, all engineering efforts must jointly consider known adversities to achieving mission success, and consistently implement proven methods to avoid, withstand, and recover from loss caused by such adversities minimizing harm to people, the environment, and other physical assets. Engineering efforts must also consider unforeseen and unknown adversities and consider flexibility and adaptability in designs holistically. The careful prioritization, consideration, and balance between desirable capabilities and potential consequences is a key LDSE aspect, as shown in the above DRM-1 mission characteristics.

A harmonizing specialty engineering disciplines need example is when a transport vehicle attaches to a space station. Safety engineering typically emphasizes a semi-active state with many sensors monitoring potential vehicle threats to frequently inform the station of the vehicle’s health and status. In contrast, reliability engineering typically emphasizes a vehicle hibernation state and few (if any) vehicle sensors
powered during hibernation, with vehicle interface health and status monitoring accomplished by space station. LDSE typically emphasizes a balance between safety and reliability optimizing overall impact on the space station and the transport vehicle.

But what if some harm level is inevitable? The next few design reference missions examine LDSE characteristics for off-nominal conditions, intending further harm minimization.

**DESIGN REFERENCE MISSION 2 (DRM-2): ILL/JINJURED CREW MEDICAL DELIVERY TO HOSPITAL** (Daniher and Cureton 1992) (Fraser 1990)

This mission builds on DRM-1 and assumes a vehicle attached to the station and is in a safe stand-by state. Therefore, if a station crew member or members become critically ill or injured—beyond the station’s medical facility capability—then the vehicle can act as a ‘space ambulance’ delivering the patient(s), plus at least one medical attendant, from the station to an Earth medical facility. This scenario’s driving requirements include:

- Delivering the ill/injured crew from the station to a ground-based or ship-based medical facility within 24 hours from undocking from the station without further harming the patient(s), therefore the vehicle must land within near proximity of the medical facility
- Vehicle ingress and securing potentially immobilized patient(s) (for example, broken spine)
- May require “special handling” during re-entry and landing such as lower deceleration and mitigating sudden maneuvering
- Attending medical assistant(s) attention is primarily on their patient(s), but attendant(s) may not qualify as a pilot or be available to manually control the vehicle flight, other than to designate the medical facility as a landing target

This mission presents a classic LDSE prioritization dilemma: balancing prioritizing mission urgency for safely delivering ill/injured crew to the medical facility (implying a landing in very close proximity) versus prioritizing safe landing without threatening people or facilities on Earth (implying a landing at a more-distant but controlled landing zone within reasonable transport distance). The difference also has significant design and cost implications: a landing in close proximity to a medical facility requires relatively precise landing control (winged vehicles or rocket-based descent and landing), whereas a landing at a distant landing zone allows for more uncertainty in the landing control (parachute use).

**DESIGN REFERENCE MISSION 3 (DRM-3): CREW RETURN TO EARTH AFTER SPACE STATION EMERGENCY** (Daniher and Cureton 1992) (Fraser 1990)

This mission builds on DRM-1 and assumes a vehicle attached to the station and is in a safe stand-by state. Therefore, if a station emergency occurs—beyond the station’s safety capabilities—then the vehicle can act as a ‘Lifeboat’ delivering the crew from the station to Earth. This scenario’s driving requirements include:

- Evacuate Space Station due to major fire, explosion, severely contaminated or leaking atmosphere, major solar flare or other radiation event, geo-political events (such as war)
- Get into vehicle and detach within 2 minutes
- Assured lifeboat vehicle accessibility requires multiple vehicles in case fire/explosion blocks access to a single vehicle
- If multiple vehicles, then cannot guarantee how many or which crew is in which vehicle, so any one crewmember must operate a vehicle—any space station crew combination (pilots, mission specialists, scientists, or passengers of any capability and culture, including potentially a range of genders, ages, languages, and skills), which requires full automation after emergency landing zone designation
- The space station emergency may damage lifeboat vehicle(s), or they may have to maneuver around station debris—may require manual vehicle automation intervention or control
- Vehicle(s) may have to loiter on-orbit up to 24 hours to obtain proper orbital positioning before descent to an emergency landing zone, and loiter on the surface up to 2 hours before ground/sea transportation vehicle arrival

This mission presents another classic LDSE prioritization dilemma: balancing prioritizing mission urgency for safe crew delivery somewhere on Earth (implying landing zone uncertainty) versus prioritizing safe landing at a prepared site without threatening people or facilities on Earth, implying landing at specific zone(s). The difference also has significant design and cost implications: crew manually controlling any vehicle entry, descent, and landing portion requires additional hardware and software to leverage flexibility and creativity in crew actions and adaptive systems to overcome unexpected adversity in space and during atmospheric entry, descent, and landing (Madni and Sievers 2018) (Sowe et al. 2016).

Other related manned space missions may be worse-case scenarios demonstrating LDSE value, for example crew retreating to safety after space station emergency, stranded crew space rescue, long-duration missions, and high frequency-of-use missions.

**CONCLUSION**

LDSE characteristics and value are evident in the above hypothetical Manned Space Rescue Vehicle discussion. Applying LDSE to major incident (disaster) response systems and national medical emergency systems (such as pandemic preparation, planning, mitigation, response, logistics, and recovery) can gain similar value. LDSE benefits include harmonizing diverse engineering disciplines to achieve resilient, safe, reliable, and secure systems with deliberate prioritization, consideration, and balance between desirable capabilities and potential consequences such as harm to people, the environment, and valuable physical assets.

**REFERENCES**


**ABOUT THE AUTHOR**

Mr. Kenneth L. Cureton serves as an adjunct lecturer in the systems architecting and engineering program at the University of Southern California and serves as the INCOSE Resilient Systems Working Group chair. Mr. Cureton retired from The Boeing Company (heritage Rockwell International) after 29 years of experience and accomplished another 16 years of successful technical leadership in commercial and government sectors. He served as the avionics team lead and later as the chief systems engineer for Rockwell’s Assured Crew Return Vehicle team, from whence this article draws operation detail. Mr. Cureton has a BS in physics from California State University Los Angeles and a MS in systems architecting & engineering from the University of Southern California.
INTRODUCTION

Loss-driven systems engineering is an emerging systems engineering subdiscipline addressing the possible losses associated with a system, including its development, use, and sustainment (Brtis and McEvilley 2020). These losses are those resulting in a system not successfully meeting performance expectations or violating stakeholder constraints in the system’s development, use, or sustainment. This focus commonly extends to include systems realizing, maintaining, or supporting a system of interest. For example, loss driven systems engineering seeks a system operating safely, an operational constraint, as well as preserving the confidentiality of a system analysis proprietary process.

Why a focus on loss-driven? As Brtis and McEvilley (2020) note, systems engineering has focused on largely capability driven methodologies. Loss driven specialties are commonly isolated specialties. Isolated thinking neglects commonalities and opportunities to think holistically about the systems. Moreover, the specialties’ work often begins after some completed work, missing opportunities to optimize system objectives. Loss-driven systems engineering strives to look at loss and the means to manage loss as a whole and not just its component types and consequences, just as systems engineering looks at a system as a whole and not just on its decomposition to components.

Systems Engineering is “a transdisciplinary and integrative approach to enable the successful realization, use, and retirement using systems principles and concepts, and scientific, technological, and management methods”, where an engineered system is “a system designed or adapted to interact with an anticipated operational environment to achieve one or more intended purposes while complying with applicable constraints” (Sillitto et al. 2019) (author’s emphasis added). What are the principles driving loss-driven systems engineering?

The paper’s aim is not to determine an authoritative answer, but rather to begin a discussion on principles. Examining a cross section of the specialties and their core principles generates a principle...
strawman set. These strawman principles compare to Watson’s (2019) and document interpretations and possible proposed changes adding to Watson’s work.

**AN APPROACH TO IDENTIFY LOSS-DRIVEN SYSTEMS ENGINEERING PRINCIPLES**

Principles are “accepted truths that apply throughout a discipline. These truths serve as a guide to the application of the discipline (Watson 2019).” Each principle must:

- Transcend life cycles
- Transcend system types
- Transcend context
- Inform a systems engineering worldview
- Not be a how-to statement
- Have literature support and/or wide acceptance in the profession
- Be focused, concise, and clear.

This paper assumes loss-driven systems engineering principles should fit well within Watson’s (2019). The approach taken to identify principles and suggested changes:

- Identified recognized core principle sets from a loss-driven specialty cross-section, whether they strictly meet Watson’s (2019) criteria or not;
- Filtered the principles from across specialties through meeting Watson’s (2019) candidate criteria; and
- Compared all specialty principles to Watson’s (2019) principles and determined any needed modifications or new principles (or hypotheses) in addition to the previous step.

### CORE PRINCIPLE CROSS SECTION

The principles selected for the cross-section were safety, security, and resilience. Watson’s ‘transcending the life cycle’ criteria filters out many specialty principles captured in literature; many widely accepted specialty principles deal with specific life cycle phases such as design.

#### Safety

Safety has numerous principle sources, many dealing with specific safety subdisciplines. One heavily cited article addressing safety broadly is Principles of Engineering Safety (Moller and Hansson 2008).

Table 1 summarizes Moller and Hansson’s (2008) principles meeting the Watson criteria or identified as obviously extensible to a candidate new principle or hypothesis. Proposed generalizations, adapted to Watson (2019) language style:

- C1: Systems engineering minimizes hazards
  
  The proposed ‘hazard’ definition is a system state or set of conditions, together with a particular set of worst-case environmental conditions, leading to loss (Leveson 2012). The concept within ‘inherently safe’ extends to avoid hazards where possible within cost, schedule, and other objective constraints.

- C2: Systems engineering seeks to control unavoidable hazards, including assuring transitions from one known acceptable mode or state to another.

  Generalizing “environmental control,” “controlling behavior,” “fail-safe design,” and “single failure criterion.”

- C3: Systems engineering uses proven and accepted processes, solutions, methods, and materials wherever possible

  Generalizing “proven design,” “quality,” and “standards,” where “proven” should include mathematically or scientifically proven within its scope.

- C4: The system should enable the human to prevent, minimize, and recover from loss when possible

  Generalizing “pilotability.”

#### Security

Security, specifically computer security and information security, has several principle sources. One of the heaviest cited is “The Protection of Information in Computer Systems (Saltzer and Schroeder 1975).” However, this paper uses McEvilley, Oren, and Ross’ (2016) Appendix F, which builds on a principle refinements survey since 1975 (Levin et al. 2007). Appendix F has 32 principles in three categories. Table 2 filters the Watson criteria or identifies as easily modified to a candidate new principle or hypothesis.

Additionally, the table does not include

<table>
<thead>
<tr>
<th>Principle (Independent malfunction)</th>
<th>A design criterion stating a single failure should not lead to system failure. System failure should only be possible in independent malfunction cases.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pilotability (safe information load)</td>
<td>The system operator should have access to the control means necessary to prevent failure, and the work should not be too difficult to perform.</td>
</tr>
<tr>
<td>Quality</td>
<td>Reliance on proven quality materials and constructions for system design.</td>
</tr>
<tr>
<td>Operational interface control</td>
<td>Focusing on controlling the interface between humans, systems, and other elements (cybernetics). For example, using interlocks to prevent human action having harmful consequences.</td>
</tr>
<tr>
<td>Environmental control</td>
<td>Control the environment so it cannot cause failures.</td>
</tr>
<tr>
<td>Controlling behavior</td>
<td>Controlling certain behavior types (alcohol and drug abuse, lack of sleep) by tests and audits.</td>
</tr>
<tr>
<td>Standards</td>
<td>Standardized system design, material usage, and maintenance procedures solutions. Standards may apply to all safety engineering areas.</td>
</tr>
</tbody>
</table>

Table 1. Safety Principle Candidates
many extensible principles, as some can apply or achieve ‘reduced complexity’ or other ‘higher level’ principles. The table’s third column notes where a security principle maps to a principle identified under safety.

First to note—procedural rigor suggests an improvement to C3, Systems engineering uses proven and accepted processes, solutions, methods, and materials when the process achieves the intended trustworthiness. Several security principles may be possible standalone candidates. For example, a candidate principle could restate ‘least privilege’: Any entity should have only the minimal privileges needed to accomplish assigned tasks. However, this candidate is a means to ensure meeting the C1 and C2.

Proposed generalizations

- C5: Systems engineering should strive for the simplest solutions
  - Generalizing ‘reduced complexity’ and numerous other security principles not listed.
- C6: Systems engineering produces evolvable systems likely to maintain or improve on loss-driven properties through change.
  - Generalizing ‘secure evolvability.’
- C7: Actions should trace to the entity responsible
  - Generalizing accountability and traceability. C7 would be critical to other specialties such as safety, where forensics determining accident causes and possible means to engineer avoiding future similar accidents is a desired outcome.

Resilience

Resilience uses a Scott Jackson and Timothy Ferris paper cited frequently on the INCOSE SEBOK resilience page (Jackson and Ferris 2013). The authors identify fourteen engineering resilient systems principles. Filtering criteria applied a bit more strictly here when a looser interpretation resulted in an existing captured principle. Many principles Jackson and Ferris captured were architectural and design principles, commonly less generalizable to the entire life cycle.

Previously captured reduced complexity left one principle to capture C8: Any critical task should be possible to perform in more than one way.

REEXAMINING WATSON’S PRINCIPLES

Watson’s principles 13 and 14 (Watson 2019) are “Systems engineering integrates engineering disciplines in an effective manner” and “Systems engineering is responsible for managing the discipline interactions within the organization.” Applying these two principles suggests a single principle set is the preferred outcome. Do the principles presented by Watson cover the need for loss-driven systems engineering principles? Do they
need extending? Are there gaps? Do the candidate loss-driven principles map into existing principles? Do the principles in the sources used suggest changes to or reinterpretations of Watson’s principles?

The answer is many of Watson’s principles and hypotheses are useful for loss-driven systems engineering, but some could use elaboration while others need re-interpreting. These include:

Principle 1: Systems engineering in application is specific to stakeholder needs, solution space, resulting system solution(s), and context throughout the system life cycle.

Assessing stakeholder needs should capture implicit loss-based needs. System solutions need assessing for loss and hazards leading to failure to meet stakeholder needs.

Principle 2: Systems engineering has a holistic system view including the system elements and the interactions amongst themselves, the enabling systems, and the system environment.

Loss-driven itself is about a holistic view to the specialties. Applying this principle to candidate principles C1 and C2 suggests hazards need such holistic views, consistent with what is in Leveson (2012).

Principle 3: Systems engineering influences and receives influence from internal and external resources, political, economic, social, technological, environmental, and legal factors.

Note this requires considering loss events associated with systems engineering itself. The various internal and external factors may result in systems engineering itself. The various internal and external factors may result in security, safety, or other incidents impacting stakeholder assets or engineer’s safety.

Principle 4: Both policy and law must have proper understanding to not overly constrain or under constrain the system implementation.

This is especially critical with many loss-driven disciplines, such as safety and the cybersecurity as a security subset.

Principle 5: The real physical system is the perfect model of the system.

This principle must extend to the physical system’s environment. How well modeled the environment and the interactions are is critical to how well the system withstands adversity in the real world.

Principle 6: A systems engineering focus is a progressively deeper understanding of the interactions, sensitivities, and behaviors of the system, stakeholder needs, and its operational environment.

Progressively deeper understanding includes bounding the unknowns better. Human space travel safety and security in contested environments are just two examples where unknowns generally always exist. Engineering for these unknowns, such as including margin, is necessary; many safety principles not discussed earlier deal with margin for the unknown and unanticipated.

Sub-Principle 6(a): Mission context definition based on understanding the stakeholder needs and constraints including bounds (or lack of) on the unknown.

Suggested change. Alternatively, perhaps a more liberal constraints’ interpretation. Constraints, such as cost and schedule, exist on systems engineering efforts, as well as limits on the ability to understand the system’s potential environments. Some long-lived systems such as power grid systems did not fully anticipate the connection to cyberspace. While arguable if this was a failure to anticipate or simply unknowable, either way future efforts must recognize future mission context knowledge constraints.

Sub-Principle 6(b): Requirements and models reflect the system and its potential environment understanding.

Suggested change. Requirement and models must understand and reflect the environment and its hazards, disruptions, and adversities. This extends beyond just the operational environment and includes all life cycle environments. For example, poor manufacturing environmental control may result in defective components and requirements and models must anticipate it. Another example is a malicious actor may attempt to inject malware during software development. Requirements and models must consider all the environments.

Sub-Principle 6(h): Understanding the system degrades during operations if not maintaining system and environmental understanding.

Suggested change. Operational and other environments change over time. Malicious actors may change intent, skill, and motivation; operational culture may evolve. Environment evolution may have numerous impacts on emergent behaviors such as safety, security, and resilience.

Principle 7: Stakeholder needs can change and system life cycle must account for changes.

Stakeholder tolerances for safety and security risk historically show change over time. For example, automotive manufacturers who may have initially accepted a risk may later issue recalls and alter vehicles following related accidents tracing to a design feature or specific vehicle component. Data breaches with a competitor may result in a company installing new firewalls to prevent a similar incident.

Sub-Principle 11(c): Systems engineering models the system and its environments.

Suggested change. Understanding system hazards requires understanding the environment.

Sub-Principle 11(d): Systems engineering designs and analyzes the system within its environments.

Suggested change. Systems engineering needs to analyze the environments to understand the possible hazards.

Sub-Principle 11(e): Systems engineering tests the system including (simulated) adversity.

Suggested change. ‘Blue sky’ testing is insufficient to assure a system meets stakeholder needs in contested and challenging environments.

DO ALL LOSS-DRIVEN SPECIALTIES REMAIN NECESSARY?

Successfully identifying a common core principle set may raise the question “do any specialties become unnecessary?” While this effort examined only three specialties in detail, initial conclusion based on analyses behind this work: No, we eliminate none, but may practice many at inseparable stages.

For example, practicing minimizing hazards (candidate principle 1) across the life cycle appears necessary regardless of specialty. On the other end, examining causalties of concern resulting in realizing a hazard and such a realization would require specialties—the security specialist for malicious causes, the reliability expert for component failures. Training is another area requiring many specialists. Requirements elicitation uses many similar analyses for data, but specialists will interpret and write the requirements specific to a specialty while others are ‘joint’ requirements.

CONCLUSIONS AND NEXT STEPS

Watson’s principles and hypotheses, with minimal additions to 1) consider the both the operational system and systems engineering process environments and 2) to address unknowns, seem overall well-suited for loss-driven systems engineering, but fail to ‘absorb’ the candidate principles. Eight potential principles to either add to Watson’s or to consider as loss-driven systems engineering principles are:
Candidate principle 1: Systems engineering minimizes hazards.

Candidate principle 2: Systems engineering seeks to control unavoidable hazards, including assuring transitions from one known acceptable mode or state to another.

Candidate principle 3: Systems engineering uses proven and accepted processes, solutions, methods, and materials when the process achieves the intended trustworthiness.

Candidate principle 4: The system should enable the human to prevent, minimize, and recover from loss when possible.

Candidate principle 5: Systems engineering should strive for the simplest solutions.

Candidate principle 6: Systems engineering produces evolvable systems likely to maintain or improve on loss-driven properties through change.

Candidate principle 7: Actions should trace to the entity responsible.

Candidate principle 8: Any critical task should be possible to perform in more than one way.

This paper looked at a loss-driven specialty subset—more thorough examinations may yield a need to modify this principle set. A workshop or technical exchange meeting, perhaps as part of a larger loss-driven systems engineering conference, is worthwhile to establish a final principle version. Additionally, examining architecture and design principles for the specialties should yield a common set for loss-driven architecture principles, perhaps as part of the same conference.

We advise an additional workshop, informed by the principles’ workshop(s), examining specialty convergence, and identifying where they remain distinct.

REFERENCES


ABOUT THE AUTHOR

Mark Winstead is the systems security engineering department chief engineer at The MITRE Corporation. After graduating with a PhD in mathematics, Mark has worked as a mathematician, software engineer, network modeling and simulation specialist, systems analyst, and systems engineer, as well as off and on as a systems security engineer, likely to retirement now. In recent years, his interests are in advancing systems security engineering by leveraging practices from other specialties such as safety.
Harmonizing the Domains of Loss-Driven Systems Engineering

Keith D. Willett, Keith.Willett@incose.org
Copyright ©2020 by Keith D. Willett. Published and used by INCOSE with permission.

ABSTRACT
System characteristics include what it is (structure, state), what it does (function, behavior), where it resides (environment, containing whole), what it uses (resources, energy source, raw material), what it contains (content), and why it exists (value delivery). An adversity produces a disturbance that can induce stress in a system so it may suffer some loss within one or more of these characteristics. Loss-driven systems engineering (LDSE) is an approach to address systemic loss in all forms helping ensure value delivery. LDSE domains include reliability, sustainability, survivability, risk management, resistance, resilience, agility, safety, and security which all work in harmony to avoid, withstand, and recover from loss. Traditional systems engineering treats these as separate domains with varying degrees of detail, rigor, and results. LDSE proposes consolidating these domains for a comprehensive, cohesive, and consistent approach to address system loss. This paper establishes interrelationships among the LDSE domains to harmonize role, fit, function, and impact among the domains focusing on sustaining value-delivery.

KEYWORDS: Loss-driven systems engineering, risk management, safety, security, agility, resistance, resilience, reliability, sustainability, survivability.

INTRODUCTION
To achieve value delivery, a system performs functions to produce desired results. To sustain value delivery while undergoing adversity, the loss-driven systems engineering (LDSE) domains contribute to system viability and relevance. LDSE describes an approach to address all forms of loss. Initially, LDSE domains include reliability (consistency), sustainability (renewable, waste management), survivability (continued existence), risk management (loss probability), resistance (retain desired status), resilience (regain desired status), agility (dynamic adaptation), safety (accidental loss), and security (malicious loss).

Viable and Relevant
Viable is capable of working successfully; being effective, efficient, and elegant. For example, we want our clean water supply to remain consistent, our food supply to remain plentiful, and the plane within which we fly to remain airborne until we reach our destination. Relevant is appropriate to current interest, or current order conformance. Absent of any adversity, the current order may change thus defining new desires. To remain relevant, a system may need to adjust the value it delivers and delivery method. LDSE helps ensure viability and relevance.

Context
Expressing a system’s meaning and value, and expressing what constitutes loss and the loss degrees may vary according to context. For example, a commercial airplane is aluminum (structure), its function is flying (behavior), and its purpose is transport people and cargo (value-delivery). LDSE provides the lexicon and method to consistently and cohesively express loss, the loss degree, and how to address loss in various contexts such as structure, behavior, content, and value-delivery.

ELABORATING ON SYSTEM-OF-INTEREST LOSS
An action sequence has a chronology of results: impact, effect, and consequence. Impact is one object forcibly contacting another. Effect is a first-order result of contact. Consequence is the importance or relevance. For example, the pool stick strikes the cue ball (impact) which moves from its current location and knocks the eight ball into the side pocket (effect) which wins the game (consequence). Impact may be a literal contact or a virtual contact. The former is some hard contact (physical) where the latter is soft (psychological or cyberspace). A cyberspace attack includes bit flows (electrons) causing a virtual impact. An impact result may also either be virtual or literal. The former includes data exfiltration (confidentiality loss), data modification (integrity loss), or data destruction (availability loss). In cyber-physical systems, malicious electron manipulation may cause a physical explosion resulting in loss of property or life.

The system of interest (SoI) may suffer a direct or indirect loss from a recent encounter (impact), a recent change resulting from an encounter (effect), or an implication from its inability to produce desired results (consequence). Distinguishing loss nuances
is important when considering system assurance (SoI focus) and mission assurance (focus on the SoI’s containing whole or that which motivates the need for the SoI) such as tactical versus strategic impact, effect, and consequence.

Impact types include:

- **Disclose**: losing intellectual property or other sensitive information negatively affecting competitive posture
- **Modify**: change to one or more system characteristics
- **Loss of X**: X∈(overall system functionality, system access, system); system does not work at all, losing virtual system access, losing physical system access, or system destruction
- **Theft/loss**: possession loss either via malicious or accidental act
- **Misled**: suffering from deceit; conclude a thought or perform an action based on falsehood
- **Loss of effectiveness**: cannot perform intended purpose; system still active but cannot produce one or more intended results
- **Compliance driver violation**: system is or acts in some manner incompatible with legal authority, regulation, policy, or some other authoritative requirement
- **Deplete**: misdirect resources or consume resources unnecessarily or without authority; use up a resource, produce excessive waste, incur unnecessary cost
- **Denial**: lack of accountability
- **Defile**: spoil the environment

Impact degrees include:

- **Destroy**: end the SoI ability to produce desired results
- **Disrupt**: temporarily incapacitate the SoI ability to produce desired results
- **Degradation**: deteriorate the SoI ability to produce desired results
- **Deny**: block access (physical); claim non-performance (opposite of non-repudiation)
- **Distort**: modify desired form (physical or virtual (data, information))
- **Deceive**: cause the SoI to perceive and thus respond to something not true thus having it produce desired results under false pretenses
- **Dated**: the SoI does not provide the features and functions available from newer alternatives; or, the SoI does not fulfill current stakeholder desires

Table 1. LDSE Domain Descriptions

<table>
<thead>
<tr>
<th>Domain</th>
<th>Description / Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reliability</td>
<td>Consistency for system characteristics; dependency.</td>
</tr>
<tr>
<td>Sustainability</td>
<td>Resource management, environment management, waste management, using renewable resources versus depletable resources.</td>
</tr>
<tr>
<td>Survivability</td>
<td>Continue to exist; remain compatible with the current order.</td>
</tr>
<tr>
<td>Risk management</td>
<td>Predicts the loss probability. Related to all LDSE aspects.</td>
</tr>
<tr>
<td>Resistance</td>
<td>Regain some desired status for system characteristics.</td>
</tr>
<tr>
<td>Resilience</td>
<td>Regain some desired status for system characteristics.</td>
</tr>
<tr>
<td>Agility</td>
<td>Dynamic adaptation; adaptable processes (development), adaptable solutions (systems), and adaptable workflows (operations).</td>
</tr>
<tr>
<td>Safety</td>
<td>Addresses accidental loss (not exclusively).</td>
</tr>
<tr>
<td>Security</td>
<td>Addresses malicious loss (not exclusively).</td>
</tr>
</tbody>
</table>

**LOSS-DRIVEN DOMAINS**

Every engineered system has a purpose to fulfill its mission such as satisfying stakeholder desires. SoI efficacy is its capacity to fulfill its mission. LDSE provides for features and functions to safeguard the SoI, preserve its efficacy, and enable the SoI to fulfill its mission. Table 1 describes the current set of LDSE domains.

**Reliability** describes a system or component’s ability to function under stated conditions for a specified period (IEEE 1990). Reliability as a measure is a failure probability. Concepts related to reliability include consistency, repeatability, durability, dependability, trustworthy, reproducibility, and lacking unintended variation. Reliability engineering includes design features helping the engineered system provide consistent and repeatable results.

**Sustainable** design, as defined by the US General Services Administration website, seeks to reduce negative impacts on the environment. Sustainability engineering designs or operates a system so they use energy and resources at a rate not compromising the natural environment or future generation ability to meet their own needs (Valerio and Brasier 2008). Sustainability measures include maximizing renewable resource use and minimizing depletable resource use.

**Survivability**, defined by Dictionary.com, is the ability to continue in existence or use. System survivability is the system’s ability to minimize a finite disturbance impacts on value delivery (Richards et al. 2007, slide 10). The system achieves survivability through either satisfying a minimally acceptable value delivery level during and after a finite disturbance, or reducing a disturbance’s likelihood or magnitude (Richards et al 2007, slide 10). An a *posteriori* survivability measure is survival rate. An indirect measure is on survivability contributors (fault-tolerance) and inferring a survivability level. An a *priori* survivability measure is the degree to which it is compatible with the current order.
Risk management predicts the loss probability (occurrence) and the loss degree (severity) across all system characteristic aspects. Loss may be real (physical) or virtual (data). There may be asset access loss, asset use loss, or asset loss. The risk posture captures stakeholder loss tolerance (risk tolerance).

Many notable engineers advocate for proactive resilience. “Resilience Engineering looks for ways to enhance the ability of organizations to monitor and revise risk models, to create processes that are robust yet flexible, and to use resources proactively in the face of disruptions (Dekker et al 2008).” “In a world of finite resources, of irreducible uncertainty, and of multiple conflicting goals, safety is created through proactive resilient processes rather than through reactive barriers and defenses (Woods and Hollnagel 2006).” LDSE captures the proactive (before something occurs), reactive (after something occurs), active (dynamic adjustment), and passive (static) spirit across resilience and resistance concepts.

A system is resistant if it produces desired results at or above a minimal efficiency threshold while preventing the effects of an adversity; resistance retains desired state, function, resources, environment, content, and value-delivery. Resistance enables the SoI to fight through the attack by preventing adverse effect(s). Prevention may avoid or withstand. There may be active resistance or passive resistance; when under missile attack, a military airplane may maneuver out of the way and deploy anti-missile devices, both are active avoidance. The airplane’s fuselage may resist flak penetration from anti-aircraft fire, a passive resistance or withstand.

A system is resilient if it produces desired results at or above a minimal efficiency threshold while undergoing the effects of an adversity; resilience regains desired state, function, resources, environment, content, and value-delivery (note: regain does not necessarily mean return to original). Resilience enables the SoI to fight through the attack by dealing with an adverse effect via withstand or recover. Withstand minimizes the adversity effects or contains the adverse effect. Recover is to achieve value-delivery even if doing so with alternative means and performing at diminished efficiency.

Agility implies dynamic adaptation versus a static adaptation where the latter includes fault-tolerance in redundancies; if the primary hydraulic system fails, the system uses the built-in secondary hydraulic system. If the secondary hydraulic system fails and we somehow install a cable system on-the-fly to maintain control, this is dynamic adaptation or agile. An agile-system or an agile-workflow adapts to sustain value-delivery in predictable and unpredictable change (Dove 2014). This implies the ability to change SoI characteristics such as structure, state, function, or resource consumption.

To be safe, according to Merriam-Webster’s online dictionary, is to be free from harm or risk; or to be unhurt. To be secure, according to Merriam-Webster’s online dictionary, is to be free from danger or free from risk of loss. Engineers often use the terms interchangeably though we intuitively have distinctions in mind. For example, we think of a seatbelt more in safety terms and a door lock more in security terms. For harmonizing LDSE domains, safety predominantly addresses accidental loss and security predominantly addresses malicious loss.

**LDSE Domain Harmonization**

Harmony is an emergent order; “harmo-
Harmony is a relational term which entails diversity and difference (Sundararajan 2013, p.2). "Harmony resides in a reality to be created each and every time (Sundararajan 2013, p.2).” Harmony is a holistic perception, an overall sense of things rather than focusing on any particular thing (Lu 2004). Harmony is a dynamic equilibrium (The Doctrine of the Mean 1971). The following narrative harmonizes LDSE domains with respect to a system providing value-delivery. From this narrative we can begin discerning LDSE domain roles, fits, functions, and impacts on each other, the system to which they apply, and establish a framework to discern their holistic relationships, find their dynamic equilibrium, and their emergent order.

Expressing value varies among stakeholders; there are differences in stakeholder currency; stakeholder currency to a politician is votes, a scientist is knowledge, a general is lives, and a banker is money. The system’s main goal is providing value-delivery in stakeholder relevant terms. Two macro-level system sub-goals are remaining viable and relevant. Measurable objectives sustaining viability and relevance are for the system to be effective, efficient, and elegant. Measurable sub-objectives to these include reliability (consistent, dependable), sustainability (renewable), and survivability (compatible with the current order); and, there are other sub-objectives at this layer (future discussion).

Methods include tactics, techniques, and procedures (TTP’s) to achieve the objectives. Risk management is a method to predict the loss probability and the loss severity to help the stakeholders determine their risk tolerance in turn driving what to do about the risk. An adversity poses a loss risk to one or more system characteristics. If the loss occurs, it occurs to some degree of adverse effect. Resistance methods attempt to retain system characteristics (avoid or withstand adverse effects). Resilience methods attempt to regain system characteristics (withstand or recover from adverse effects). Resistance and resilience forms vary among agile (dynamic, composable), static (passive, playbook), proactive (preemptive), and reactive (responsive).

Methods invoke products and services (solutions) as part of their processes. With respect to LDSE, these solutions are safeguards addressing safety (accidental loss) and security (malicious loss). Safety and security products and services provide the solution space helping ensure viability and relevance so the system continues to provide value-delivery.

From this narrative, we see reliability, sustainability, and survivability as measurable objectives. Risk management, resistance, resilience, and agile are methods to achieve the objectives. Safety and security provide solutions the methods invoke. The LDSE domains are necessary but not sufficient to sustain value-delivery. LDSE is part of a larger construct (future discussion) for the system to achieve and sustain value delivery.

### Toward LDSE Principles

Table 2 provides thoughts toward LDSE principles; incomplete and for discussion. Thoughts toward refining LDSE principles include resilience types and ethics. Resilience types:

- **Innate:** born with; applies to natural living systems; not contrived by humans
- **Inculcated:** developed through experience, training, education, and practice
- **Inherited:** passed on genetically (e.g., height, eye color)
- **Internal:** within an individual; not external
- **External:** outside an individual; not internal

### Table 2: Thoughts Toward LDSE Principles (continued)

<table>
<thead>
<tr>
<th>Domain</th>
<th>Notional Principles</th>
</tr>
</thead>
<tbody>
<tr>
<td>Agile</td>
<td>Adapt to predictable change&lt;br&gt;Adapt to unpredictable change&lt;br&gt;Adapt predictably (deterministic); playbooks&lt;br&gt;Adapt unpredictably (non-deterministic) or flexibly; emergent behavior</td>
</tr>
<tr>
<td>Security (Willett 2008)</td>
<td>Confidentiality: ensure only authorized disclosure&lt;br&gt;Integrity: ensure only authorized modification&lt;br&gt;Availability: ensure ready for use; ensure no service denial&lt;br&gt;Possession: ensure physical retention; ensure no physical loss or theft&lt;br&gt;Authenticity: ensure conformance with reality; ensure no deceit&lt;br&gt;Utility: ensure fit for purpose&lt;br&gt;Privacy: right no observation, the right to forgetting&lt;br&gt;Non-repudiation: ensure accountability for actions; ensure non-deniability&lt;br&gt;Authorized use: ensure only authorized [cost-incurring] service use</td>
</tr>
<tr>
<td>Safety</td>
<td>Minimize unintentional harm; minimize intentional harm&lt;br&gt;Sacrifice property before life&lt;br&gt;Sacrifice non-human life before human life&lt;br&gt;Safeguard Sol’s state, function, resource, content, environment, value delivery&lt;br&gt;Safeguard other Sol’s hierarchy on harm degree choices in preference order (priority):&lt;br&gt;- Avoid rather than deflect (no contact)&lt;br&gt;- Deflect rather than damage (light contact, redirecting force)&lt;br&gt;- Damage rather than destroy (medium contact)&lt;br&gt;- Destroy rather than kill (hard contact)&lt;br&gt;- Kill only as a last resort</td>
</tr>
</tbody>
</table>
**Inherent**: essential, intrinsic; applies to non-living systems, contrived by humans; resilience emerges via the normal SoI features and functions

**Planned**: a contrived SoI part, intentional design such as redundant component. Redundant feature or function. Invoke something known (playbook)

**Emergent**: an agile behavior invokes planned features and functions in a manner producing new behaviors; composing a new ability producing desired results

Harm may be necessary for strategic success: the pawn to save the king, and win the battle or sacrifice the data server to learn more about adversary strategy. Intentional harm will at times be necessary to resolve moral dilemmas; autonomous vehicle *must*choose to hit four school children on the left, a woman pushing a baby carriage on the right, or crash into the barrier straight ahead thus causing harm to itself and its contents. Accepting this takes us down the path that a SoI perpetrating some harm is necessary. Now comes the extremely difficult question to the acceptable degree of harm and in what form the harm remains acceptable. This will vary according to context such as cultural differences in morality, and acceptable behavior and consequences. The final version of safety principles must capture these concepts.

**CONCLUSION**

LDSE domains work in harmony providing a comprehensive approach to identify and integrate loss-driven requirements in a holistic solution design addressing all system state, behavior, resources, content, environment, and value characteristics. LDSE facilitates producing the risk posture reflecting stakeholder loss tolerance. LDSE complements opportunity-driven systems engineering as iterative methods to sustain viability and relevance to achieve the main value-delivery goal (Willett 2020).

**REFERENCES**


**ABOUT THE AUTHOR**

Dr. Keith D. Willett is a senior strategist and enterprise security architect for the United States Department of Defense with over 35 years’ experience in technology. He is a co-chair for the INCOSE working groups on Systems Security Engineering and Agile Systems and Agile Systems Engineering; plus, an active member in working groups for Resilient Systems and Systems Science. He is the lead for INCOSE’s Future of Systems Engineering (FuSE) agility project with intent toward systems engineering methods developing and operating inherently adaptable systems.
SPECIAL
FEATURE
DECEMBER  2O2O
VOLUME 23/ ISSUE 4

1. INTRODUCTION

One LDSE challenge, applying to all projects involving multiple specialties and disciplines, is siloism. According to Investopedia (2020), “a silo mentality is a reluctance to share information with employees of different divisions in the same company.” Therefore, in a project context, siloism is individual reluctance to share information with different specialty experts on this project. In any project, it may be a phenomenon reducing multiple specialty effectiveness when acting jointly. This attitude reduces the organization's efficiency and, at worst, contributes to a damaged corporate culture. In LDSE context the expectation is all project specialties will respect all other specialties on the same project, leverage their strengths, and harmonize their processes.

This does not make LDSE more vulnerable than any other project to siloism. However, since LDSE is a project type, siloism mitigation precautions should be the same as any other project especially those involving many systems engineering specialty areas.

The following discussion first describes siloism causes and consequences. Secondly, it describes a frequently suggested siloism treatment, namely, employing the integrated product team (IPT) concept on projects. According to Heckler’s work (2000), "IPTs have become a cornerstone to success in our programs today… They pierce barriers such as culture, functional issues, political agendas, personal problems, and physical distance between stakeholders.”

a) Siloism Causes

The basic siloism cause is humans are often reluctant to share information with other specialists on a project. Their reluctance’s root cause goes beyond this paper's scope. But a general agreement is this reluctance often exists. Factors encouraging information sharing include organizational structure and management involvement. These factors exist in the IPT concept discussed below.

b) Using Multiple Systems Engineering Specialty Areas

Achieving a quality product in an LDSE environment requires many systems engineering specialties. Example systems engineering specialties include reliability, safety, risk, security, cyber, mechanical, electrical, and many more. Using these systems engineering specialties generates the need for a method to mitigate siloism.

2. INTEGRATED PRODUCT TEAMS

In the systems community one commonly used methodology identified to mitigate the damaging siloism effects is the IPT (integrated product team). According to the Defense Acquisition University (2009) an IPT is a “team composed of representatives from appropriate functional specialties working together to build successful programs, identify and resolve issues, and make sound and timely recommendations to facilitate decision making.” An IPT will contain the specialties listed in the previous section.

An IPT is an organizational concept in which multiple specialists working within a single organizational unit, the IPT, are less likely to make suboptimal decisions without considering larger consequences. Typical errors without the IPT may be conflicts in requirements or overlaps in requirements. These errors still may occur, but the IPT environment will make them less likely. According to Browning’s work (2009, 1416), “the intent of an IPT is to integrate diverse individual perspectives at the lowest level possible.”

Browning (2009, 1401-24) further explains, there are other organizational concepts, such as the design structure matrix (DSM), but the IPT has its own

Scott Jackson, jackson@burnhamsystems.net
Copyright ©2020 by Scott Jackson. Published and used by INCOSE with permission.
features providing an advantage. Of these approaches, the IPT offers the most effective specialty area use possibility.

So, an IPT has the following three primary features: First, it should represent the specialties considered essential to project success. Second, it should have leadership dedicated to assuring all the specialties work together and share information. A key IPT leader responsibility is helping the discipline members recognize the IPT goal gaining all IPT members’ cooperation regarding achieving a balance in the specialty decisions.

The IPT’s third feature is it should focus on a single element (the IPT “product”) of the system in question. Browning (2009, 1415-17) explains how cross-cutting specialties can assign across elements. Browning (2009, 1415-17) further explains an IPT approach advantage is its easy integration with the work breakdown structure (WBS) and the integrated mater plan (IMP). Because it focuses on a single product element, the IPT approach promises to us no more than the minimum specialty area number.

**THE IPT ORGANIZATIONAL ROLE**

When employing an IPT organizational approach, the organization breakdown structure (OBS) will be the same as the product breakdown structure (PBS). Each product element will correspond to a single organizational element, or group. In addition, Blanchard and Fabrycky (2006) describe other approaches, such as the functional matrix approach and the pure project organization.

The IPT organizational approach can benefit the project to which it applies. In this view an entire system, for example an aircraft, can be a “product.” As shown in Figure 1, an aircraft program can comprise several IPTs, each one representing a major sub-system. For example, there may be an avionics IPT and a propulsion IPT. This organization’s advantage is the avionics IPT and the propulsion IPT may have different specialties. So, grouping them together with their subsystems will allow the specialty groups to work together more closely. Any IPT membership should include representatives from potentially impacted specialties. As such, all IPTs should be cross-functional as needed. There are often discipline teams participating in IPTs to provide representation on those teams, without necessarily dedicating individuals as IPT members. There may also be cross-functional IPTs, as shown in Figure 1. For example, if every team requires a cost specialist, there may be a single cost IPT. This is what the IPT concept brings to the table, first an organizational structure and a managerial imperative creating the cooperative environment implementing all the specialties with minimal conflict. In this concept, the primary responsibility lies with the project manager to achieve specialty cooperation.

**3. SUMMARY**

In short, one IPT approach value is it minimizes damage caused by conflicting or duplicated specialties and disciplines. Care must assure (1) the IPT leader takes steps to leverage technical specialties to balance priorities and avoid conflicts and suboptimization, (2) the overall team includes the correct IPT organizations, and (3) each IPT includes representation from the minimum specialty number contributing directly to the sub-system in question.

---

**REFERENCES**


**ABOUT THE AUTHOR**

Scott Jackson is an independent researcher and consultant specializing in the systems approach to commercial aircraft development. He is an INCOSE fellow and an ISSS (International Society of the Science of Systems) member. His research interests include the resilience of engineered systems and advanced aspects of decision management. His degrees include an MS from UCLA and a PhD from the University of South Australia. He taught systems engineering and systems architecting at the University of Southern California for ten years. He has written four books on systems engineering, the systems approach, and systems architecting.
Systems Engineering the Conditions of the Possibility
(Towards Systems Engineering v2.0)

Keith D. Willett, kwillett@ctntechnologies.com
Copyright ©2020 by Keith D. Willett. Permission granted to INCOSE to publish and use.

ABSTRACT
Traditional systems engineering’s focus is on cause and effect. When we turn a wheel, pull a lever, or flip a switch we expect a certain outcome. This is a rules-based approach where stimulus-response is deterministic in a well-defined, well-bounded, finite, and predominantly static system. If anything deviates from the expected, simple systemic structures (logic gates) or simple rules (if-then-else) provide optional preplanned action. Human intervention provides the intelligence and action necessary for dynamic adjustment to a negative event (adversity, avoid loss) or detecting and dynamically adjusting to a positive event (opportunity, seek gain). The now and future discipline of systems engineering (systems engineering v2.0) has the tools to transcend cause-effect and effectively embrace the nondeterministic, flexibly defined, blurred-boundaries, highly combinatorial if not infinite, and adaptability. Systems engineers can design solutions to adapt to predictable and unpredictable change for the system to remain viable while encountering adversity (loss-driven) and relevant when threatened by obsolescence (opportunity-driven). In addition to cause and effect, systems engineering v2.0 is systems engineering the conditions of the possibility.

This paper does not intend to provide answers, but provides a framework for discerning better questions and eliciting research in the many technical areas providing continual dynamic adaptation of complex socio-technical systems of systems. Realizing systems engineering v2.0 will come from the hard work of many over years. We are already on the way with this being one more step toward formalizing a new discipline.

INTRODUCTION
In keeping with Immanuel Kant’s philosophy, conditions are the context necessary for a range of predictable and unpredictable outcomes (possibility). Nature provides many conditions; space is a condition for realizing three dimensional objects. As we move forward into systems engineering’s future, engineers provide conditions for continual dynamic adaptation of complex socio-technical systems of systems including numerous predictable and unpredictable outcomes for the system to remain viable and relevant under nominal and adverse situations.

Appendix A provides a notional diagram for a systems engineering v2.0 framework starting with the context within which the system of interest (SoI) finds a role, fit, and function. The SoI may be a subset of people, process, technology, or environment (containing whole (ecosystem) systems of systems), the SoI is one system in a workflow of dynamic interactions where trigger events (when) prompt people (who) to perform a process (how) using technology (what) within an environment (where) to produce results for consumption to bring about a desired outcome (why). Workflows have goals. The primary goal of all systems is provide value-delivery under nominal conditions; a goal of some systems is sustain value-delivery under adverse conditions. Function-driven, loss-driven, and opportunity-driven strategies sustain goals; objectives are measurable steps within strategies. Methods provide the tactics, techniques, and procedures to achieve objectives. Risk management is one method with two supporting methods of resistance (retain status; avoid, withstand) and resilience (regain status; recover). Methods invoke solutions including those for safety and security. The sections below provide concepts, assumptions, and details for the systems engineering v2.0 framework and set up future research to expand this new discipline’s structure and content.

Systems engineering v2.0 evolves traditional systems engineering as a discipline to produce and sustain SoI’s as complex adaptive socio-technical systems-of-systems. This evolution includes modifying some current thinking about engineering sub-disciplines (computer engineering, software engineering) and adding new sub-disciplines (cybersecurity engineering, artificial intelligence (AI) engineering, quantum engineering, and cognitive engineering). Appendix A provides a notional diagram for a systems engineering v2.0 framework purposely intending to standalone, it does not find immediate alignment to the V-model. The motivation for this is to establish (or not) the role, fit,
function, and purpose of systems engineering v2.0 on its own not as an extension of current practice, but as something necessary in itself. Then, if it has merit, find alignment to current practice for appropriate integration and extension.

Traditional systems engineering shortcomings include assumptions for a correct or optimal solution, unambiguous requirements are possible, sequential development process, engineers do what utility value determines they should do, systems are decomposable, that a centrally controlled development process exists, static system context, human factor ignorance, system behavior is determinist cause-effect, and context-independent solutions (Pennock and Wade 2015). Additional systems engineering v1.0 shortcomings include solutions as software platforms, fragility to cyberspace threats, and autonomous features and functions. Systems engineering v2.0 addresses these shortcomings by adding design methods for adaptable optimality, fuzzy or blurred requirements, ongoing recursive development, and accommodating patterns as abstract standard intermediate forms from which to create and sustain complex systems.

Many traditional systems like airplanes, automobiles, transportation systems, and medical devices are now software platforms hosting many computer processors and software applications to enhance mechanical operations and provide system functionality, functional exchanges (communication), and other features and functions like safety, security, agility, resistance, resilience, reliability, sustainability, and survivability. This software contributes to system interconnectivity on a scale unprecedented in human history, the emerging Internet-of-things (IoT). Artificial intelligence (AI) provides for autonomy throughout observe, orient, decide, act, command, and control (OODA+C2) far more than just autonomous action.

The intangible, virtual cyberspace interweaves with tangible, real mechanical devices to create cyber-physical systems where manipulating electrons on a wire has physical effects, robots, process control (industrial control systems), transportation management, and autonomous vehicles. Successful operation implies greater efficiency in time and resource expenditure (energy consumption). Failure to adapt to an adverse situation (bad weather) or malicious manipulation (state-sponsored cyberattack) may result in loss of life and property.

The fabric of everyday life across social and industrial domains ever increasingly entwines with technology. Technology is a medium through which we engage in social interactions and from which we obtain news and education with which we make life decisions. Technology is fundamental in commercial exchanges, banking and financial management, and healthcare including robotic surgery and medical device operation. In coming decades, we will grow to depend on autonomous technology like drones, autonomous vehicles, and robots traversing public streets as well as the hallways at work, school, and shopping malls.

Cyberspace is not inherently bad or good, it is a medium through which bad or good things may happen. Human nature remains constant. For bad actors to perpetrate a bad act, cyberspace removes the need for physical proximity, reduces the time to act, and amplifies a single bad act to potentially a global scale; cyberspace is a force multiplier. Systems engineering v2.0 helps design solutions aware of and adaptable to sustain confidentiality, integrity, availability (ready for use), possession (anti-theft), utility (fit for purpose), authenticity (anti-deception), privacy, non-repudiation (anti-deniability), and authorized use (anti-misallocation of cost incurring service) among many other considerations found within the systems engineering v2.0 framework.

Systems engineering is the multi-disciplined approach to help manage human transition into this socio-technical age of symbiosis between people and technology in a vastly interconnected world. Systems engineering v2.0 will help explicitly design features for emergent behavior, adaptation to change, produce nondeterministic results, and help superimpose many dichotomies (Figure 1) that seem paradoxical in a single system, but reflect the complex adaptive socio-technical systems-of-systems reality.

1.1 SOCIO-TECHNICAL AGE

The industrial age evolved into the digital age (or information age) which is evolving into a vastly interconnected world. We are entering a socio-technical age of complex symbiotic relationships among people and technology. We are integrating technology ever deeper in our everyday lives. The internet-of-things digitizes and interconnects our appliances, door locks, televisions, computers, cars, and our phones. Our refrigerators know how much milk we drink and ensure we never run out via automated ordering and aerial drone delivery. Our vehicles are autonomous and have responsibility for detecting and resolving adverse conditions including new and unique conditions for which the unprepared vehicle must discern the situation, determine the best action, and perform that action.

Systems dealing with this complexity require explicit design to deal with the predictable and the unpredictable; to deal with interconnection and symbiosis. Our increasing dependence on such technology requires trusting the technology to perform those actions we desire while keeping us safe from accidents and secure from malicious intent.

1.2 CONDITIONS OF THE POSSIBILITY

In his work Critique of Pure Reason, Immanuel Kant describes conditions of the possibility as “a necessary framework for the possibility of a given list of entities, space is a necessary [natural] condition for the existence of cubes (Kant 1929).” Space does not cause cubes nor does it guarantee cubes will exist; rather, space is a dependency for the possibility of cubes. Systems engineering v2.0 provides a framework to identify conditions necessary for realizing systems that sustain viability and relevance when encountering predictable and unpredictable change. Systems engineering v2.0 will help identify natural conditions and help develop engineered conditions. Just like the natural condition of space does not cause nor prescribe the existence of cubes, engineered conditions do not cause nor prescribe a system’s viability or relevance; rather, systems engineering v2.0 establishes the conditions of the possibility. We build engineered conditions into the system, its containing whole, and its environment (ecosystem) as states of ableness. The conditions provide states within which reside the ability to act on internal and external forces providing the stimuli prompting the system to act in order to realize the possibility (Figure 2).

In Kant’s example, space does not act, it merely is. Given a stimulus for creating cubes, some Sol may create cubes (a
possibility) in part because space exists (a condition). In this sense, space is a necessary but not sufficient condition for the existence of cubes. Assuming space exists as a condition, cubes may never exist. Creating space is not a wasted effort, because space is still a condition for possibility of spheres. Similarly, systems engineering v2.0 may facilitate creating conditions for which we never realize the intended possibility. Systems engineering v2.0 may anticipate the possibility of cubes and create the necessary conditions. However, reality never gives us cubes but gives us something we never anticipated, spheres. And it turns out spheres are exactly what we need. The lack of realizing cubes is not a failure; rather, preparation for realizing alternatives, spheres, is a success.

Conditions are not quite potential because potential connotes some preplanned or known state, behavior, or resource. Conditions are not constrained to the known and do not constrain the known; rather, they provide states of ableness so we may preplan actions (playbook) or we may engage in previously unforeseen actions producing unforeseen results (dynamic composition).

Systems engineering v2.0 helps the designer prepare for the known and the unknown. This implies the system may be or do something for which we did not explicitly plan. This is part of preparing the conditions. Yes, we want the system to adapt as necessary. No, we do not want the system to adapt to be or to do something unintentionally harmful or destructive. This prompts the encoding of axioms for appropriate action based on some moral, ethical, and/or legal framework for dilemma and conflict resolution represented in adjudication rules and adjudication logic. Defining appropriate action is context dependent, an acceptable moral framework will vary by culture. How we determine liability, those to whom we assign liability, and those who accept liability will have influence on this adjudication framework.

For example, one adjudication framework axiom may be do no harm. If we encode such an axiom, our system could never win a chess game, which requires tactical sacrifice to gain strategic advantage to win the game. Perhaps minimize intentional harm and minimize unintentional harm are better axioms. These acknowledge the need for the SoI to do some harm under certain conditions. Now comes the challenge of determining the acceptable degree of harm. We take a step down a very slippery slope representing real-life quandaries requiring representation in systems engineering v2.0.

1.3 ASSUMPTIONS

Systems thinking in systems engineering v2.0 includes analysis thinking, synthesis thinking, transcendence thinking, and temporal thinking. Analysis thinking decomposes the system of interest (SoI) into its constituent parts, analyzes the parts, and then recompiles the parts into the SoI with the intent of better understanding the SoI by understanding its parts. Synthesis thinking identifies the containing whole for the SoI; identifies the SoI role, fit, and function as a member of the containing whole, and analyzes the SoI impact on the containing whole, how the containing whole acts or becomes different by virtue of the SoI. Transcendence thinking identifies and analyzes emergent SoI properties and its containing whole. For example, decomposing humans into their constituent parts helps to understand their biology; however, it does nothing to explain the emergent property of emotion (love). Temporal thinking orients on states, behaviors, and resources over time, change, stasis, delay, and amplifying small changes over time.

A system is a set of constituent parts (structure) interacting (behavior) to take input (resources) to produce a benefit (value) to other systems and/or its containing whole and environment and/or its contents. A system is a bounded structure. A system has constituent parts. Distinct from its constituent parts, a system may have contents, cargo, people, or data. A system may be in a state (a particular condition of parts, features). A system may perform behavior (functions, functional exchanges). A system may possess or have access to some resource (input, energy (fuel)). A system resides within an environment and a system may reside within a containing whole, part of a system of systems.

A system may be real or virtual, nature-made (natural) or human-made (engineered). For example, a railroad system is real, human-made; the respiratory system is real, nature-made; a government system is virtual, human-made; and, an ecosystem is real, nature-made. An engineered system is one created by people. A natural system exists as part of the emergence of nature’s current order. The current order may be natural (an ecosystem) or contrived (a government system or the Internet). The system most compatible to the current order, survives. The current order defines the benchmark for determining if a system is viable and relevant. A viable system is capable of working successfully; a relevant system is appropriate to current needs.

Both engineered systems and natural systems provide benefits to other systems, they deliver value. A system having purpose is a human quality as is a system producing desired results. All engineered systems serve a purpose to the humans creating them. Natural systems may serve a purpose to humans as well. Natural systems provide benefits to other natural systems in an equilibrium conforming to the current order. Natural systems may benefit from other systems, but these other systems do not serve a purpose to natural systems for which they provide benefit.

A system may refer to a thing (computer system) or a process (system for betting on horse races). A person may be a system; a collective of people may be a system or a system-of-systems. The systems engineering process is itself a system. What applies to a system applies to the systems engineering process. There is some concept of system that transcends our direct experience. We can define some systems in terms of empirical evidence, we can see a system, hear it, taste it, smell it, and/or feel it. What about mathematical systems? We can see the representations of such systems in a generally agreed upon nomenclature, but we do not directly experience mathematical systems through our sensory perception.

Defining system in general terms is problematic because any one definition finds difficulty in capturing the entirety of what it is (state), what it does (behavior), and what it has (resources). If a system is a set of somethings, then that which is not decomposable is not a system but may be the smallest part of the system in which it resides. Atoms are systems consisting of neutrons, protons, and electrons. Protons are decomposable into quarks which are themselves not decomposable; and electrons are not decomposable. By this definition, quarks and electrons are not systems. Some quantum computing architectures isolate protons and electrons and manipulate them to store and process data. In this sense, protons and electrons are part of a quantum computing system. If the definition of system is something that responds to a stimulus, then electrons are systems because they respond to the stimulus within the quantum computer.

We may define a system by what it is (state) or what it does (behavior). What a system is describes its physical and virtual makeup, such as parts, wholes, and emergent properties (hardware and software, body and mind). What a system does describes its functions and functional exchanges. A system’s value is less in what it is and what it does, and more in the results it produces. For example, an airplane is aluminum. Its function is to fly. Its value is transporting people and cargo. Losing an airplane is less in the replacement cost (what it is) as compared to the loss of benefit from transporting people and cargo (its value). Therefore, we may also define a system by the value it delivers.
Traditional systems engineering focuses on structure and function. Systems engineering v2.0 adds the value perspective and attempts to build in the ability to sustain value-delivery while encountering adversity; threats to SoI viability and relevance. The system itself cannot handle all contingencies; too expensive to equip and encode in all systems. The containing whole also fails to handle all contingencies but can handle many more contingencies than any constituent system alone. The containing whole may orchestrate alternative states, behaviors, and resources for its constituent systems to continue to produce benefits. Systems engineering v2.0 helps build in intra-system and inter-system relationships for adaptability to sustain continual value-delivery.

Any given system must act independently to produce its desired results and submit to governance by the containing whole to adapt to change. Systems engineering v2.0 helps to establish the conditions for the possibility of independence, subordination, substitution, modification, and permutation with the intent of being adaptable to predictable and unpredictable change.

A means to an end helps achieve a goal. A goal or an end is another term for purpose and applies to humans but not natural systems. In social systems, Immanuel Kant’s categorical imperative holds true, treat others never merely as a means to end, but always at the same time as an end. Engineered systems and natural systems may treat other engineered and natural systems as purely means to produce a benefit. For example, human cells regenerate themselves on various timelines. The cell is not an end; the cell is a means to renew the human tissue or a major organ like the liver. Some systems are means and not ends. Some systems are expendable and have short-term relevance and limits on viability.

### 2 Framework for Systems Engineering: The Conditions of the Possibility

Systems engineering v2.0 allows system designs to integrate into the vast socio-technical age operating environment. Traditional systems engineering produces results achieving a state and a certain functionality. Systems engineering v2.0 embraces ongoing design to sustain value-delivery. Sustaining value complements achieving states and functionality as both are necessary, for example a strategy to get rich (risk seeking; maximize upside) may differ from a strategy to stay rich (risk averse; minimize downside).

When designing a socio-technical age SoI, we define goals for the SoI in particular context, why the SoI exists. Strategies describe how to sustain the goals. Objectives are the measurable steps to sustain the strategies. Methods are the actions to pursue the objectives. Entities have characteristics and exist in an environment. The environment provides or contains enablers and constraints for the entities’ ability to perform methods. Trigger events from various trigger sources (actuators) provide stimuli prompting an entity to act.

Stakeholder requirements remain the primary driver behind systems design. Systems engineering v2.0 may propose additional requirements given the system’s role, fit, function, and desired impact (value) within its containing whole and as part of the socio-technical age operating environment. Systems engineering v2.0 accommodates socio-technical age features and functions as part of explicit systemic design and operation. Some functions are algorithmic (rule-based) and some are axiomatic (principle-based). Systems engineering v2.0 provides a structure and method to capture the requirements and guide system production to remain viable and relevant in the socio-technical age. Systems engineering v2.0 includes:

- **Context**: the circumstances forming the setting in terms for understanding, assessing, and analyzing; and, for expressing meaning and value

- **System Characteristics**: a standard framework for what comprises a system; details defining and describing a system; system parts we may address to provide desired results and sustain viability and relevance

- **Workflow Taxonomy**: a standard framework for expressing workflows collectively comprising operations

- **Goal**: a broad primary outcome; why the SoI exists; goals to sustain value-delivery

- **Strategy**: approach to achieve and sustain a goal; this is part of the segue from why to what stakeholders want to how to fulfill stakeholder needs

- **Objective**: a measurable step to sustain a strategy; this elaborates on and quantifies the details of getting from why to what to how

- **Method**: a systematic approach to achieving objectives

  - **Tactic, technique, procedure**: actions to pursue an objective; this is how to achieve and sustain an objective; also may be a process

  - **Method details include agile (dynamic, composable), static (passive, playbook), proactive (preemptive), reactive (responsive)**

- **Entity**: something real or virtual to perform methods; this includes what (technology) and who (people)

- **Characteristic**: a quality belonging to an entity

- **Resource**: something an entity possesses or accesses, materials, things (real or virtual), raw material (inputs), fuel (energy to operate)

- **Environment**: geography, location, or facilities within which the SoI will be (state), do (behave), or have (resource)

- **Enabler**: makes value-delivery possible

- **Constraint**: limits or restricts [successful] value-delivery

- **Trigger**: activity initiator (actuates), state, time, location, event

- **Trigger source**: trigger provider; an activity initiator (actuator)

The systems engineering v2.0 framework accommodates both the systems engineering process and solution the systems engineering process produces. The framework is highly flexible and accommodates systems engineering v2.0 concepts addressing systems engineering v1.0 shortcomings. The framework helps define and capture requirements for the conditions and does not address what tools accomplish the conditions. The conditions are the state of ableness to achieve the possibility, the solution prepares to be adaptable, agile, resistant, resilient, and to remain viable and relevant.

Systems engineering v2.0 applies new technologies as part of its tool repository and methods to develop solutions, but does not advocate for applying those tools which is up to the designer. New technologies provide constructs of how that enable the realization of systems engineering v2.0 and thus are part of the systems engineering v2.0 tool repository. Engineering practices pre-late 1800’s did not include electricity or electrical engineering because they did not exist. Similarly, computers circa 1945 were the size of large rooms and ran on vacuum tubes. Aeronautical engineering and automotive engineering did not include computers because they were not practical. New technologies enabling systems engineering v2.0 to encode conditions of the possibility include advancements in classic probability, quantum probability, quantum decision theory, artificial intelligence, machine learning, set-based design, mathematical category theory, continuous decision-making, real-time reliability-based optimization, and real-time orchestration engines. Many new technologies are distinct disciplines requiring explicit expertise, such as cybersecurity engineer, cognitive engineer, computer engineer, social engineer, quantum engineer (quantum algorithms), and software engineer. Many disciplines provide details to expand the systems engineering v2.0 framework including complexity engineering, systems science, viability theory, social choice theory, and quantum decision theory. Modeling and simulation
become part of operations for continual exploration and optimal choice selection; the same conditions tomorrow may result in a different choice than today because of a shift in context (non-deterministic).

2.1 ELABORATING ON THE CONTENTS OF THE SE V2.0 FRAMEWORK

Appendix A provides a notional systems engineering v2.0 framework diagram. Throughout the framework are notations of who, what, why, when, where, and how as six interrogatives capturing atomic-level elements about a SoI and building compounds constituting SoI details, providing dynamic compositionality building blocks. SoI atomic elements include:

- **Who**: goals; related concepts include vision
- **What**: processes; related concepts include methods; tactics, techniques, and procedures (TTP’s); strategies; and objectives
- **Why**: materials, things; related concepts include solutions, tools
- **When**: roles; responsibilities; related concepts include socio-cultural, individuals, teams
- **Where**: environment, location, facilities; related concepts include containing whole and current order

The following details elaborate on the systems engineering v2.0 framework contents. Note reality is fungible, value-delivery may take on many forms and occur by various substitutions. The framework is for binning concepts and organizing thoughts and does not necessarily imply hard and fast delineation lines. Many rules provide for the predominance and not exhaustiveness of a concept. Systems engineering v2.0 recognizes many rules have exceptions and we need enough exceptions before we modify the rule and create new rules. Any single framework aspect provides many future research opportunities to elaborate on the details, quantification, modeling, and encoding systems engineering v2.0 interactions orchestrating dynamic adaptation for continual optimization sustaining value-delivery.

**Context** frames meaning and value. A context change may change what constitutes value, thus prompting a change in the system’s value-delivery or the terms expressing value-delivery. Future research includes formalizing a standard context ontology.

**System characteristics** are structure/state, behavior/function/functional exchange, content, resource, environment (current order, containing whole), and value-delivery. The characteristics require formal ontologies and interactions throughout.

**Workflow taxonomy** is a trigger event prompting people to perform processes using technology within an environment producing results for consumption bringing a desired result. A collection of systems and workflows comprise operations. Dynamic adaptation includes changing systems and changing workflows to sustain value-delivery.

**Goals** express what systems engineering v2.0 helps sustain in the systems it produces. These goals do not supplant stakeholder requirements, but supplement stakeholder requirements for systems in the vastly interconnected world of the socio-technical age. The primary SoI goal is value-delivery. Another primary goal is sustaining value-delivery, continuing to deliver value in nominal and adverse operating conditions. Two additional goals provide for sustaining value-delivery: the system shall remain viable and the system shall remain relevant. A viable system is capable of working successfully; a relevant system is appropriate to current needs. A viable and relevant system can survive or remains compatible with the current order (ecosystem).

The **strategies** to sustain the goals are function-driven, loss-driven, and opportunity-driven. Function-driven strategies include system functions and functional exchanges addressing effectiveness, the system is doing what it should be doing. Loss-driven strategies address the negative side of risk and include avoid, withstand, detect, defend, respond, restore, and recover. Opportunity-driven strategies address the positive side of risk and include seek, embrace, predict, preempt, cause, and achieve to sustain advantage and optimization.

**Function-driven strategies** sustain the system’s ability to be (state), to do (behavior), and to have (resources) via functions and functional exchanges necessary for the system to remain effective. Traditional systems engineering predominant focus is on functions. Systems engineering v2.0 adds to these and includes function-driven as well as loss-driven and opportunity-driven strategies.

**Loss-driven strategies** sustain the system’s viability; resistance and resilience. Loss-driven resistance strategies retain viability and proactively adapt to avoid or withstand an adversity or adverse effect. Loss-driven resilience strategies regain viability and reactively adapt to respond, restore, and recover from an adversity or adverse effects. This loss-driven systems engineering (LDSE) concept looks at what can go wrong, avoids what can go wrong, responds to something gone wrong, and becomes better at dealing with what can go wrong in the future. The determination of going wrong involves some undesirable change in elegance (resource levels and resource consumption), efficiency (performance parameters), and effectiveness (binary). LDSE domains include risk management, safety, security, agility, resilience, reliability, sustainability, and survivability.

**Opportunity-driven strategies** sustain the system’s relevance; explore and innovate. Opportunity-driven strategies retain relevance and proactively adapt to avoid or withstand obsolescence, proactively adapt to a new value-delivery model. Opportunity-driven strategies regain relevance and reactively adapt to recover and restore relevance, reactively adapt to a new value-delivery model. Opportunity-driven strategies seek, embrace, cause, achieve, sustain, and optimize the system’s ability to be effective, efficient, and elegant.

The pursuit of opportunity in SE v2.0 seeks to satisfy unmet desire or need that is known, unknown, explicit, implicit, predictable (deterministic), or unpredictable (non-deterministic). Opportunity-driven systems engineering (ODSE) domains include innovation (explore/experiment, exploit), predict/proactive/preemptive, discover/react, evolve, contingency planning, and tradeoff analysis.

The ODSE concept looks at what can go right, seeks what can go right and what can go better, respond to improvement opportunities, and becomes better at capitalizing on future opportunities (improves its ability to improve). Innovation seeks better. If the innovation process determines the SoI is doing OK, it continues to exploit the status quo. If the innovation process determines the SoI is not doing OK, it explores a better way. The exploration process may wait for a problem (reactive) or seek alternatives to become better or develop contingency plans (proactive). Exploring for better includes SoI self-experimenting, add/delete/modify state (condition), behavior (functions, functional exchanges), and resources for continual relevance and viability. Formal innovation methods may emerge in systems engineering v2.0 under the innovation engineering discipline.

SoI design accommodates the need to be effective differently. Imposing an additional stakeholder requirement in a new feature or function, the system can accommodate such an addition, modification, or deletion. This relates to the system being conducive to exogenous change or exogenous-driven adaptation. SoI design seeks to be more effective. Given some calculation that determines the system can be more elegant (modify/optimize resource consumption), the system can seek a new state (to be), behavior (to do), or resource (to have) for such optimization. A negative change in available resource levels prompts rationing...
resource consumption or a priority and preemption scheme allocating resources to essential activities. Similarly, if there is an increase in SoI’s seeking the same resource, rationing may occur locally to optimize the whole. Given some calculation determining the system can be more efficient (modify/ optimize performance), the system can seek a new state (to be), new behavior (to do), or new resource (to have) for such optimization. For example, an environment change may prompt performance level adjustments.

SoI objectives are the quantifiable, measurable steps to achieve and sustain a strategy. This elaborates on and quantifies the details of getting from why to how. Systems engineering v2.0 uses three macro-level objectives: effective (produces desired results), efficient (performance), and elegant (resource expenditure). The effective objective is a binary measure of the SoI producing desired results or not producing desired results. This is a value-oriented measure. Is the SoI producing the desired value regardless of its state (condition) or behavior (function) or available resource. The efficient objective measures the SoI producing desired results within specified performance parameters with the continual goal to optimize performance. Measures include enumerating performance parameters, establishing efficiency thresholds, monitoring performance, and providing awareness in terms similar to green zones (good), yellow zones (approaching not good), and red zones (not good). The elegant objective measures the SoI producing desired results with minimal resource expenditure. Measures include resource repositories (current levels), resource availability, and resource expenditure. Resources include people, time, money, fuel, raw material, data, information, and knowledge.

For example, in ODSE context, effective seeks to produce new results. Efficient seeks/embraces/causes/achieves/sustains/optimizes the ability to produce desired results within specified performance parameters (continual performance optimization). Elegant seeks to minimize resource expenditure, embrace minimal resource expenditure, cause a resource expenditure shift, and achieve a desired resource expenditure level (continual resource expenditure optimization). Elegance includes minimize waste, minimize depletable resource use, maximize renewable resource use, and maximize sustainability.

SoI objectives consist of measurable states, behaviors, and resources. In socio-technical systems, states include physical, cognitive, and mechanistic structure. Behaviors include team interaction, social interaction, workflow, human-machine interface, cognitive assistants, and mechanistic operation. Resources include inputs (raw material), what runs the system (electricity, food, fuel), and what keeps the system running (money).

The measure expressions are not static; the stakeholder value measures may vary. For example, stakeholder currency for politicians is votes, for scientists is knowledge, for a military general is lives, and for a banker is money. The point is, the stakeholder value expression may vary according to context. Future research in a decision support framework will address the need for collect-once and reuse-many where the same data provides for multiple value-delivery expressions.

The effective, efficient, and elegant collective* provides a viability measure. The effective, efficient, and elegant collective also provides a relevance measure. Are the collectives equal? No. They measure different things under the same categories. Therefore, as we continue to decompose systems engineering v2.0 and it’s focuses, we need to distinguish the viability components versus the relevance components. *The term collective is an abstract reference to some mathematical relationship among effective, efficient, and elegant. The mathematical relationship may be a sum or product involving weights, priorities, confidence levels, accuracy, age of last details, and other factors; and, it may vary according to context.

There are many sub-objectives to effective, efficient, and elegant, for example reliability, sustainability, and survivability.

- **Reliable**: produce desired results consistently; produce desired results within specified deviation limits from expected; related concepts include consistency, repeatability, durability, dependability, trustworthy, and reproducibility
- **Sustainable**: reduce negative impacts on the environment [https://www.gsa.gov/real-estate/design-construction/design-excellence/sustainability/sustainable-design, last accessed 6-Sep-2019]; minimize depletable resource use and maximize renewable resource use
- **Survivable**: remains viable and relevant (persists) in nominal and adverse operating conditions; remain compatible with the current order

SoI methods are actions to pursue an objective and include TTP’s, and processes; this is how to achieve and sustain an objective. To remain viable, engage in X actions to address loss. To remain relevant, engage in X actions to address opportunity. Each method category has its own domains and dynamic relationships:

- **Function-Driven Systems Engineering**: focus on value-delivery
- **Loss-Driven Systems Engineering**: focus on sustaining value-delivery with predominant focus on sustaining viability; avoid, withstand, and recover from loss
- **Opportunity-Driven Systems Engineering**: focus on sustaining value-delivery with predominant focus on sustaining relevance; seek gain, contingencies for continual optimization
- **Risk Management**: predicts the loss (probability) (occurrence) and the loss degree (severity) across all system characteristics. Loss either occurs or it does not which leads to two methods addressing loss:
  - **Resilient**: produces desired results at or above a minimal efficiency threshold while preventing the effects of an adversity; retain state, behavior, resource; avoid loss
  - **Resistant**: produces desired results at or above a minimal efficiency threshold while undergoing the effects of an adversity; regain state, behavior, resource; handle loss
- **Agile**: produce desired results in a predictable and unpredictable change environment; active, dynamic, composable; more flexible, but slower
- **Static**: produce desired results in a predictable change environment; passive, pre-established responses; playbooks; less flexible, but faster
- **Proactive**: predict and preempt; anticipate, cause
  - Implies predictive analytics
  - Inductive logic to reason forward from cause to effects, conditions to possibility, means to end
  - Abductive logic reasoning backward from effects to necessary and sufficient cause, possibility to necessary and sufficient conditions, and end to necessary and sufficient means
- **Note**: classic probability based on Kolmogorov’s axioms provide deterministic predictive analytics for events and states; quantum probability based on Dirac-von Neumann axioms provide non-deterministic analytics for modeling cognitive processes (quantum decision theory or quantum cognition) and other purposes; systems engineering v2.0 extends the classic probability (deterministic) use to include quantum probability (non-deterministic)
- **Reactive**: detect and defend; respond, react, recover
  - Implies analytics to identify, describe, and explain (deductive and abductive
Methods invoke available *products* in the form of non-person entities and socio-technical *services* potentially including people. Products and services are the *solutions* (tools), safeguards. A safe system can produce desired results while minimizing harm to the SoI, its contents, and surroundings; addresses *accidental* adversity. A secure system can produce desired results within specified risk tolerance limits; addresses *malicious* adversity. Security solutions provide the ability to harden, protect, defend, attack, and exploit. Solutions for SDSE are not yet determined. Solutions are entities with characteristics and consume resources to optimize value-delivery.

Entities perform methods and include technology and people, real and virtual. People may include individuals, teams, groups, organizations, societies, or nations. Technology is any tool enhancing human performance, such as a pencil, hammer, vehicle, computer, and artificial intelligence. Socio-technical combines people and technology and some elements are unique to socio-technical settings, joint cognitive systems and cognitive assistants. A real entity is tangible, a person or computer. A virtual entity or a virtual entity part is intangible, cognitive (mind), software, data.

**Characteristics** are qualities belonging to an entity and include states and behaviors. A *state* may include a structure, feature, attribute, or configuration representing a *condition*. A *behavior* may include a function or functional exchange. A functional exchange is some internal communication (endogenous) or external communication (exogenous); the former implies an *intraconnection* state and the latter implies an *interconnection* state. Behaviors may be exogenous; externally induced change, external change (provided a patch), directed from without where the SoI is subordinate to command and control from a higher order. Behaviors may be endogenous; self-induced change, self-changing, directed from within where the SoI may be autonomous. Exploration methods may include *recursive self-design* (continual improvement) and *agency* (encoding the ability to act producing a particular effect vs engage in particular functions).

**Resources** are something the entity possesses, may possess, or may access and are necessary to sustain a desired state or behavior; materials and things (real or virtual, tangible or intangible). Resources include *knowledge, skills, and efficacy*. *Knowledge* comprises structured details about something including context, problem, and solution. *Skills* are the ability to use one’s knowledge effectively, efficiently, and elegantly. *Efficacy* is the belief in one’s ability. In part, efficacy comes from culturally formed beliefs and mental models and find expression in permission, restriction, and direction. Permissive includes what the SoI may be, may have, and may do. Restrictive includes what the SoI may not be, may not have, and may not do. Directive includes what the SoI must be, must have, and must do. Either by conscious effort or by default, we encode efficacy into technology, algorithmic bias. Systems engineering v2.0 includes algorithmic oversight to minimize unconscious bias and encode culturally acceptable bias.

Context drives determining what constitutes SoI optimal operation and appropriate behavior. Exploring may include change anticipation in a contingency preparation context. Adapting on-the-fly takes time. If the SoI can invoke a preplanned state, behavior, or resource, then adaptation is faster than developing from scratch. Both are necessary. The tradeoff space among effective, efficient, and elegant constrains how many resources to dedicate for contingency planning.

**Environment** is the geography, location, or facilities where the SoI is to be (state), to do (behave), or to have (resource). The SoI may inherit its environment from the containing whole or the containing whole may be distinct from the environment and SoI considerations include the containing whole and the environment. The environment is part of the context providing for the expression of meaning and stakeholder value.

**Enablers** facilitate [successful] operation. Enablers come in states, behaviors, resources, and environment. For successful operation, the SoI must be X, must do X, must have X, must reside within X, or the environment must provide X. What the environment must provide may be an infrastructure such as communication infrastructure, transportation infrastructure. **Constraints** limit or restrict [successful] operation. Constraints come in states, behaviors, resources, and environment. Successful operation curtails if the SoI is X, if the SoI does X, if the SoI has X, if the SoI resides within X, or if the environment provides or does not provide X. Constraints include legal, regulatory, contractual agreements, service level agreements, ethics, natural laws, scientific laws, state, behavior, resources, and environment. Some constraints may be self-imposed, policy. Considerations for constraints include maxims like just because we can do not mean we may (legal, regulatory, policy), just because we may (legal) does not mean we should (ethical, risk). Compliance drivers are one constraint category and include externally imposed (legislation, regulation), internally self-imposed (policy), and negotiated (contracts, service level agreements).

**Triggers** initiate an activity (actuates). Triggers include *time* (when), *state* (what, are), *behavior* (do), *resources* (have), and *environment* (where). *Time* includes any temporal-driven trigger; schedule, day/time, periodicity, roadmap. *State* includes some SoI aspects, what the SoI observes, or what some other entity observes and communicates to the SoI (the orchestration engine). There may be a state change or a state continuation beyond a time limit. A state includes the *something you know* concept as an action trigger; we know they know we know. An *event* is the occurrence of some activity; system identifies a successful security breach by a threat. **Environment** is the physical or logical proximity; an adversary in a secure space prompts a response action.

**Behavior** includes what the SoI does, what the SoI observes happening, or what some other entity observes happening and communicates to the SoI’s orchestration engine. A trigger may be a behavior change or a behavior continuation beyond a time limit. **Resource** includes something the SoI possesses or can access, observed as in possession or accessible by another SoI, or what some other entity observes as possessed or accessible and communicates to the SoI. For example, a nation state deemed less than responsible possessing nuclear capability triggers a response action. **Environment** includes geography (terrain and weather) or location (longitude/latitude); geofencing. An autonomous vehicle may act differently according to environmental factors affecting driving conditions. Vehicular activity will vary between clear weather in a rural setting at midnight vs a rainy day in an urban setting during rush hour.

A trigger source provides a trigger, an activity initiator; the actuator. Trigger sources may be exogenous or endogenous to the SoI. The SoI is subject to change; responsive to authorized change, resistant to unauthorized change. The SoI perceives some threat (trigger), the orchestration engine decides to act (actuator), and subsequently invokes some dynamic adaptive behavior.

### 3 SYSTEMS ENGINEERING V2.0 PATTERNS

Systems engineering v2.0 includes a pattern-based approach to systems engineering. By creating solutions for a socio-technical age, there is interaction and interdependence among SoI’s. This requires a consistent approach to design and operations coupled with flexibility to choose disparate solutions. Any given system may highly individualize its role, fit, function, and impact. If it hopes to integrate into world-wide structure, there must be some common fundamentals we may reuse across many systems. We may guide these
common fundamentals using patterns. Note: these common structures do not impose upon system designs; rather, common structures are available for those desiring to participate in the socio-technical age. In analogy, the Internet did not impose upon the world but made available to those who wished to participate in the digital age. Systems engineering v2.0 will help discover, design, implement, and operate similar structures for the socio-technical age.

The systems engineering v2.0 framework provides a foundation to develop and subsequently apply patterns; archetype and other reusable structure capture and reuse. The systems engineering v2.0 framework provides a structure identifying abstractions for developing architecture-patterns (agile-solutions or agile-operations), design-patterns, and decision-patterns applicable to developing and operating complex adaptive socio-technical systems-of-systems. Pattern types include:

- **System Archetypes**
  - Behavior patterns within a system and among systems; recurrent motifs in system dynamics; encode known dynamics, known problems, and provide clues to solutions
  - Causal patterns
- **Architecture-Patterns**
  - Agile Architecture Pattern for Systems (Dove and LaBarge 2014)
  - Capture and reuse system modules to compose solutions
  - Agile Architecture Pattern for Operations
  - Capture and reuse operation modules to compose workflows
- **Design-Patterns**
  - Capture and reuse development knowledge
- **Decision-Patterns** (Willett 2017)
  - Capture and reuse operational knowledge; cybersecurity decision patterns (CDPs)
- **Ecosystem Patterns**
  - One way to think of an ecosystem is an entity community existing within a physical or logical boundary interacting as a system
  - Capture and reuse ecosystem structure (entity organization), states (to be), and behaviors (to do)
- **Anti-patterns**
  - Known bad solutions to a problem in a particular context

A pattern language provides the grammar to express patterns. Pattern language includes semantics (meaning expression) and morphology (the study of patterns, their parts, and their relationships in order to express meaning). The intent is to capture standard approaches relevant to systems engineering v2.0 to help create and sustain SoIs in the socio-technical age.

In systems engineering v2.0, patterns and pattern languages will provide for algorithms and axioms throughout the complex adaptive socio-technical systems-of-systems lifecycle. These patterns will represent details for expressing, encoding, and operationally sustaining goals, strategies, objectives, methods, entities, entity characteristics, resources, environment, enablers, constraints, action triggers, and trigger sources. In part, patterns provide abstract standard intermediate forms to create and adapt complex systems. The solution development concept still exists under systems engineering v2.0. Additionally, systems engineering v2.0 facilitates solution composition from standard intermediate forms; elements, subassemblies, components, or modules.

**4 CONCLUSION**

Systems engineering the conditions of the possibility is creating the context for continual dynamic adaptation of complex socio-technical systems of systems that includes predictable and unpredictable stimulus and outcomes for the system to remain viable and relevant. The details herein propose a systems engineering v2.0 framework as a notional structure identifying many future research areas contributing to inherently adaptable system design, assembly, and operation discipline.

As shown in Appendix A, the systems engineering v2.0 framework starts with context (range of conditions) within which the SoI finds a role, fit, and function. The SoI may be any combination or subset of people, process, technology, or environment. The primary goal of all systems is value-delivery under nominal conditions; a goal of some systems is sustain value-delivery under nominal and adverse conditions. Strategies sustain goals; function-driven, loss-driven, and opportunity-driven. Objectives are measurable steps within strategies. Methods provide the processes, tactics, techniques, and procedures to achieve objectives. Risk management is one method and includes resistance (retain status) and resilience (regain status). Methods invoke solutions including those for safety and security.

Patterns provide guides for producing modules throughout the systems engineering v2.0 framework. Modules are necessary to compose systems and compose operations each in static or dynamic instantiations. The orchestration concept provides for dynamic continual adaptation with the module composition. Future research for real-time orchestration includes the individual areas and dynamics among compositionality theory, set based design, category theory, and quantum cognition.

The systems engineering v2.0 framework is not the answer, but at best begins asking the right questions to find the answers for developing and sustaining complex adaptive socio-technical systems-of-systems for the socio-technical age. The framework provides stubs for integrating systems engineering v2.0 method and practice. Most of these details do not exist today which provides research opportunities to identify and explore the possibilities. Systems engineering v2.0 is not prescriptive in the technology types to use; solutions will always change. However, systems engineering v2.0 is prescriptive in using current technologies to create a viable socio-technical age operating environment. Current is a relative term; therefore, what is current today will change tomorrow. Hence, advocating systems engineering for the conditions of the possibility providing the ability for the SoI to adapt to remain viable and relevant.

* See Appendix A: SE v2.0 Framework (Notional) on next page.

**REFERENCES**


**ABOUT THE AUTHOR** [See page 31]
Appendix A: Systems Engineering v2.0 Framework (Notional)

- **Socio-Cultural**
  - Economic
  - Institutional
  - People, desired results, value

- **Technical**
  - Technology
  - Industrial

- **Spatial**
  - Environment
  - Geography
  - Infrastructure

- **Temporal**
  - Continuous
  - Continual (periodic)
  - Interval: fixed interval event interval

- **Behavior**
  - Function
  - Function exchange
  - Process, method...

**System of Interest (SoI)** Characteristics (CH)

- **Structure**
  - State

- **Behavior**
  - Function
  - Functional Exchange

- **Content**
  - Resources

- **Environment**
  - Current Order
  - Containing Whole

- **Value-Delivery**
  - why

**Workflow Taxonomy (W)**

- **Trigger Event**
  - when

- **Process**
  - how

- **Environment**
  - where

- **Results**
  - Consumption

- **Desired Results**
  - why

**Goals (G)**

- Viable
- Relevant

**Strategies (ST)**

- Function-Driven
- Loss-Driven
- Opportunity-Driven

**Objectives (O)**

- Effective
- Efficient
- Elegant

- Reliability (consistent)
- Sustainability (renewable)
- Survivability (compatible w/ current order)

**Methods (M)**

- Function-Driven Systems Engineering (produce desired results)
- Loss-Driven Systems Engineering (avoid, withstand, recover)
- Opportunity-Driven Systems Engineering (seek gain, explore/exploit)

**Risk Management**

- (probability of loss and severity)

- Resistant (retain status)
- Resilient (regain status)

- Agile (dynamic, composable)
- Static (passive playbooks)
- Proactive (preemptive)
- Reactive (responsive)

**System of Interest (SoI) Characteristics (CH)**

- Context (CN)
- Temporal (when)
- Behavior (how)

**Constraints**

- enablers

- triggers
- triggers sources
Annual INCOSE International Workshop
29 - 31 January 2021
www.incose.org/iw2021

2020 KEY NUMBERS

170 Meetings

618h of Productive Workshop

13h of Social Events
Save the date
Register now
2021 Annual INCOSE International Workshop
29 - 31 January 2021
Join us to the first virtual workshop
www.incose.org/iw2021

Save the date
Honolulu July 17 - 22, 2021
www.incose.org/symp2021

Save the date
Call for submission
www.incose.org/hsi2021

HSI2021 Human Systems Integration Conference
San Diego, CA, USA
November 17-19, 2021