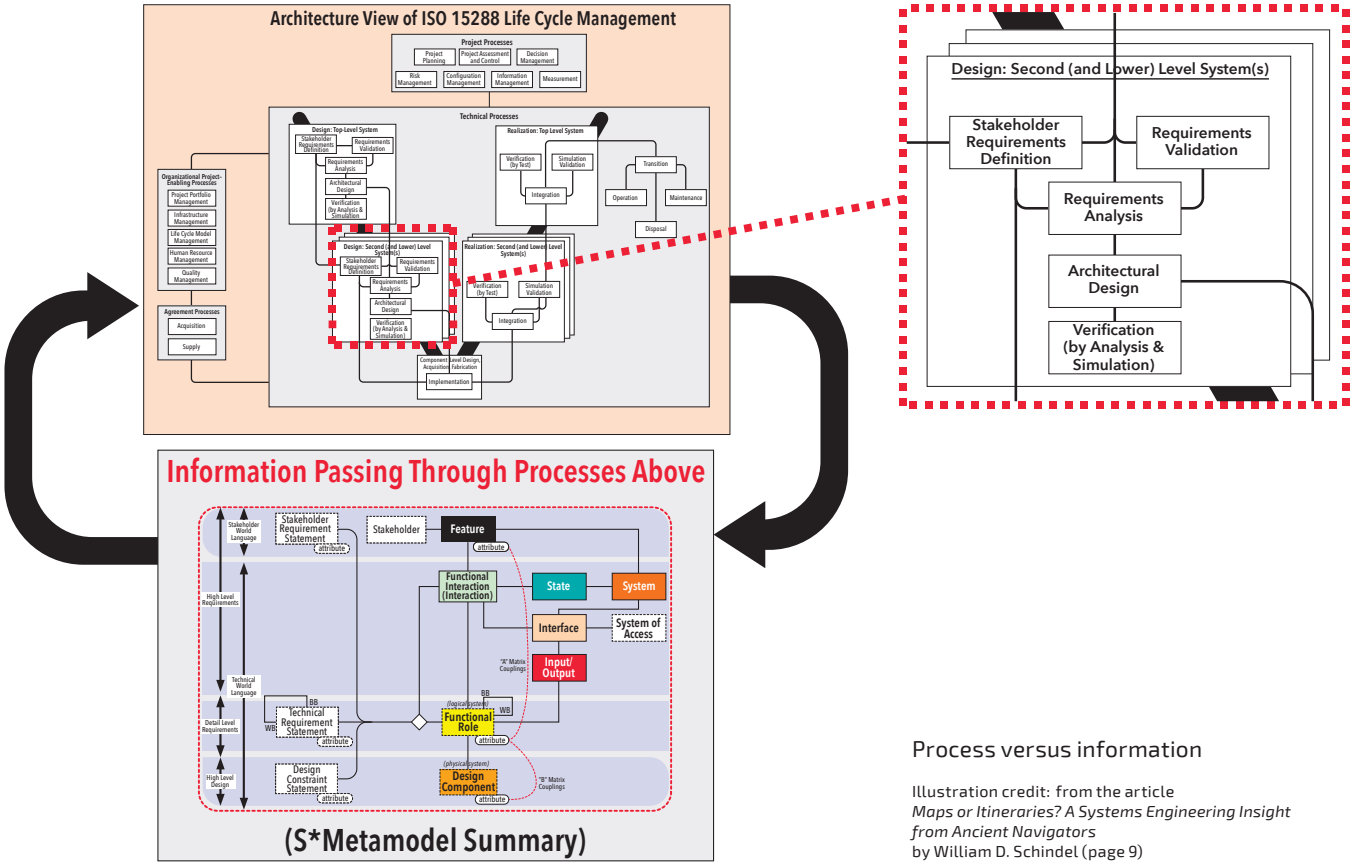
INSIGHT

This Issue's Feature:

Theoretical Foundations: Impacts on Practice I



Process versus information

Illustration credit: from the article *Maps or Itineraries? A Systems Engineering Insight from Ancient Navigators* by William D. Schindel (page 9)

AUGUST 2024
VOLUME 27 / ISSUE 4

A PUBLICATION OF THE INTERNATIONAL COUNCIL ON SYSTEMS ENGINEERING

CATIA MAGIC

DESIGN AND ENGINEER FASTER
WITH THE CLOUD-BASED
COLLABORATIVE MBSE SOLUTION

LEARN MORE

DASSAULT SYSTEMES

INSIGHT

A PUBLICATION OF THE INTERNATIONAL COUNCIL ON SYSTEMS ENGINEERING

AUGUST 2024 VOLUME 27 / ISSUE 4

Inside this issue

About This Publication

INFORMATION ABOUT INCOSE

INCOSE's membership extends to over 23,000 members and CAB associates and more than 200 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 charters 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)
INSIGHT is the magazine of the International Council on Systems Engineering. It is published six times per year and

OVERVIEW

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.

Editor-In-Chief William Miller
insight@incose.net +1 908-759-7110

Layout and Design Chuck Eng
chuck.eng@comcast.net

Member Services INCOSE Administrative Office
info@incose.net +1 858 541-1725

PERMISSIONS

*** PLEASE NOTE: If the links highlighted here do not take you to those web sites, please copy and paste address in your browser.**

Permission to reproduce Wiley journal Content:
Requests to reproduce material from John Wiley & Sons publications are being handled through the RightsLink® automated permissions service.

Simply follow the steps below to obtain permission via the Rightslink® system:

- Locate the article you wish to reproduce on Wiley Online Library (http://onlinelibrary.wiley.com)
- Click on the 'Request Permissions' link, under the ‹ARTICLE TOOLS› menu on the abstract page (also available from Table of Contents or Search Results)
- Follow the online instructions and select your requirements from the drop down options and click on 'quick price' to get a quote
- Create a RightsLink® account to complete your transaction (and pay, where applicable)
- Read and accept our Terms and Conditions and download your license
- For any technical queries please contact customercare@copyright.com
- For further information and to view a Rightslink® demo please visit www.wiley.com and select Rights and Permissions.

AUTHORS – If you wish to reuse your own article (or an amended version of it) in a new publication of which you are the author, editor or co-editor, prior permission is not required (with the usual acknowledgements). However, a formal grant of license can be downloaded free of charge from RightsLink if required.

Photocopying
Teaching institutions with a current paid subscription to the journal may make multiple copies for teaching purposes without charge, provided such copies are not resold or copied. In all other cases, permission should be obtained from a reproduction rights organisation (see below) or directly from RightsLink®.

Copyright Licensing Agency (CLA)
Institutions based in the UK with a valid photocopying and/or digital license with the Copyright Licensing Agency may copy excerpts from Wiley books and journals under the terms of their license. For further information go to CLA.

Copyright Clearance Center (CCC)
Institutions based in the US with a valid photocopying and/or digital license with the Copyright Clearance Center may copy excerpts from Wiley books and journals under the terms of their license, please go to CCC.

Other Territories: Please contact your local reproduction rights organisation. For further information please visit www.wiley.com and select Rights and Permissions.
If you have any questions about the permitted uses of a specific article, please contact us.

Permissions Department – UK
John Wiley & Sons Ltd.
The Atrium,
Southern Gate,
Chichester
West Sussex, PO19 8SQ
UK
Email: Permissions@wiley.com
Fax: 44 (0) 1243 770620
or

Permissions Department – US
John Wiley & Sons Inc.
111 River Street MS 4-02
Hoboken, NJ 07030-5774
USA
Email: Permissions@wiley.com
Fax: (201) 748-6008

ARTICLE SUBMISSION insight@incose.net

Publication Schedule. ***INSIGHT*** is published six times per year. Issue and article submission deadlines are as follows:

- October 2024 – 1 July 2024
- December 2024 – 1 September 2024
- February 2025 issue – 1 November 2024
- April 2025 issue – 2 January 2025
- June 2025 issue – 1 March 2025
- August 2025 issue – 1 May 2025

For further information on submissions and issue themes, visit the INCOSE website: www.incose.org

 ISSN 2156-485X; (print) ISSN 2156-4868 (online)

ADVERTISE

Readership
INSIGHT reaches over 23,000 members and CAB associates and uncounted employees and students of more than 130 CAB organizations worldwide. Readership includes engineers, manufacturers/purchasers, scientists, research and development professionals, presidents and chief executive officers, students, and other professionals in systems engineering.

Issuance	Circulation
2024, Vol 27, 6 Issues	100% Paid

Contact us for Advertising and Corporate Sales Services
We have a complete range of advertising and publishing solutions professionally managed within our global team. From traditional print-based solutions to cutting-edge online technology the Wiley-Blackwell corporate sales service is your connection to minds that matter. For an overview of all our services please browse our site which is located under the Resources section. Contact our corporate sales team today to discuss the range of services available:

- Print advertising for non-US journals
- Email Table of Contents Sponsorship
- Reprints
- Supplement and sponsorship opportunities
- Books
- Custom Projects
- Online advertising

Click on the option below to email your enquiry to your nearest office:

- Asia and Australia corporatesalesaustralia@wiley.com
- Europe, Middle East and Africa (EMEA) corporatesaleseurope@wiley.com
- Japan corporatesalesjapan@wiley.com
- Korea corporatesaleskorea@wiley.com

USA (also Canada, and South/Central America):

- Healthcare Advertising corporatesalesusa@wiley.com
- Science Advertising Ads_sciences@wiley.com
- Reprints Commercialreprints@wiley.com
- Supplements, Sponsorship, Books and Custom Projects busdev@wiley.com

Or please contact: Marcom@incose.net

CONTACT

Questions or comments concerning:

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

Advertising—*please contact:*
Marcom@incose.net

Member Services – *please contact:* info@incose.org

ADVERTISER INDEX

Month Volume 27-4

CORPORATE ADVISORY BOARD — MEMBER COMPANIES

Aerospace Corporation, The
Airbus
AM General LLC
Analog Devices, Inc.
Arcfield
Australian National University
AVIAGE SYSTEMS
Aviation Industry Corporation of China, LTD
BAE Systems
Bechtel
Becton Dickinson
Belcan Engineering Group LLC
BMT Canada
Boeing Company, The
Booz Allen Hamilton Inc.
Boston Scientific Corporation
C.S. Draper Laboratory, Inc.
California State University Dominguez Hills
Carnegie Mellon Univ. Software Engineering Institute
Change Vision, Inc.
Colorado State Univ. Systems Engineering Programs
Cornell University
Cranfield University
Cubic Corporation
Cummins, Inc.
Cybernet MBSE Co, Ltd
Dassault Systèmes
Defense Acquisition University
Deloitte Consulting, LLC
Denso Create Inc
DENTSU SOKEN INC
Drexel University
Eaton
Eindhoven University of Technology
EMBRAER
FAMU-FSU College of Engineering
Federal Aviation Administration (U.S.)
Ford Motor Company
GE Aerospace
General Dynamics
General Motors
George Mason University
Georgia Institute of Technology
Hitachi Energy
IBM
Idaho National Laboratory
ISAE – Supaero
ISDEFE
IVECO Group
Jama Software
Jet Propulsion Laboratory
John Deere & Company
Johns Hopkins University
KBR, Inc.
KEIO University
L3Harris Technologies
Lawrence Livermore National Laboratory
Leidos
LEONARDO
Lockheed Martin Corporation
Los Alamos National Laboratory
Loyola Marymount University
Magna
ManTech International Corporation
Marquette University
Massachusetts Institute of Technology
MBDA (UK) Ltd
Medtronic
MetaTech Consulting Inc.
Missouri University of Science & Technology
MITRE Corporation, The
Mitsubishi Electric Corporation
Mitsubishi Heavy Industries, Ltd
Modern Technology Solutions Inc
National Aeronautics and Space Admin. (NASA)
National Reconnaissance Office (NRO)
National Security Agency Enterprise Systems
Naval Postgraduate School
Nissan Motor Co, Ltd
Northrop Grumman Corporation
Pacific Northwest National Laboratory
Pennsylvania State University
Petronas International Corporation Limited
Prime Solutions Group, Inc
Project Performance International (PPI)
Purdue University
QRA Corporation
Rolls-Royce
RTX
Saab AB
SAIC
Sandia National Laboratories
Saudi Railway Company
SENSEONICS
Shanghai Formal-Tech Information Technology Co., Ltd
Shell
Siemens
Sierra Nevada Corporation
Singapore Institute of Technology
Southern Methodist University
SPEC Innovations
Stevens Institute of Technology
Strategic Technical Services LLC
Swedish Defence Materiel Administration (FMV)
Systems Planning and Analysis
Taiwan Space Agency
Tata Consultancy Services
Thales
The George Washington University
The University of Arizona
The University of Utah
Torch Technologies
TOSHIBA Corporation
Trane Technologies
Tsinghua University
UK MoD
Universidade Federal De Minas Gerais
University of Alabama in Huntsville
University of Arkansas
University of California San Diego
University of Connecticut
University Of Lagos
University of Maryland
University of Maryland Global Campus
University of Maryland, Baltimore County
University of Michigan, Ann Arbor
University Of Nairobi
University of New South Wales, The Canberra
University of Southern California
University of Texas at El Paso (UTEP)
US Department of Defense
Veoneer US Safety Systems, LLC
Virginia Tech
Volvo Cars Corporation
Volvo Construction Equipment
Wabtec Corporation
Weber State University
Woodward Inc
Worcester Polytechnic Institute (WPI)
Zuken, Inc

FROM THE EDITOR-IN-CHIEF

William Miller, insight@incose.net

We are pleased to publish the August 2024 *INSIGHT* issue published cooperatively with John Wiley & Sons as the systems engineering practitioners' magazine. The *INSIGHT* mission is to provide informative articles on advancing the practice of systems engineering and to close the gap between practice and the state of the art as advanced by *Systems Engineering*, the Journal of INCOSE also published by Wiley.

The focus of this August issue of *INSIGHT* is theoretical foundations: impacts on practice, featuring the contributions of MBSE Patterns Working Group chair and INCOSE fellow William (Bill) Schindel. Bill was asked by Sandy Friedenthal and Heinz Stoewer beginning in 2019 to provide materials from his past work on theoretical foundations for the preparation of the forthcoming *Systems Engineering Vision 2035* led by Sandy, Heinz, and Garry Roedler published in 2021 (www.incose.org/publications/se-vision-2035). Bill's contributions towards the Vision 2035 were reviewed by Tom McDermott, Chris Paredis, David Rousseau, Jon Wade, and Michael Watson (current INCOSE president-elect).

The *Vision 2035* was preceded by the *Systems Engineering Vision 2020* (2007) and *A World in Motion: Systems Engineering Vision 2025* (2014). In particular, the Vision 2025 called for stronger foundations noting that *systems engineering practice is only weakly connected to the underlying theoretical foundation, and educational programs focus on practice with little emphasis on underlying theory*. The *Vision 2025* objective was that *the theoretical foundation of systems engineering encompasses not only mathematics, physical sciences, and systems science, but also human and social sciences. This foundational theory is taught as a normal part of systems engineering curricula, and it directly supports systems engineering methods and standards. Understanding the foundation enables the systems engineer to evaluate and select from an expanded and robust toolkit, the right tool for the job*.

Bill asserts "that much of that foundation is closer than realized, not always requiring discovery 'from scratch.' There are well-established foundations of STEM and other disciplines, discovered and highly successful during three centuries of the transformation of human life. These foundations await a wider awareness and exploitation by the systems community, providing a powerful starting point for what will follow. The foundations are both quantitative and qualitative, and richly endowed with humanistic aspects." Bill summarizes three phenomenon-based elements of that foundation, providing already known starting points: the systems phenomenon, the value selection phenomenon, and the model trust by groups phenomenon." All these elements have significant implications for systems engineering practitioners, educators, and researchers. We thank Bill along with co-author Troy Peterson.

We lead the August *INSIGHT* with Bill Schindel's metaphorical thought piece questioning the approach to systems engineering as described in the INCOSE *Systems Engineering Handbook:* "Maps or Itineraries? A Systems Engineering Insight from Ancient Navigators." Processes and procedures are the heart of current descriptions of systems engineering and enterprise-specific business process models reinforce this focus on process and procedure. The attention devoted to describing process, sequence, or activity usually exceeds by orders of magnitude the amount devoted to describing the information flowing through that process. Scholarly works suggest the ancients navigated by itineraries that preceded the innovation of maps. These itineraries listed ports and landmarks to facilitate commercial and military sailing and lists of locations and distances on land routes. An itinerary is a sequence of steps whose performance is expected to move us from point A to point B. By contrast, a map describes the geographic space of interest, identifying points in geographic space and the relationships between those points. A map is a relational model that answers an infinity of questions that may arise in various situations. A map is not a procedure. This is important to understanding the current state of systems engineering. Metaphorically, systems engineers must "navigate" a type of "journey," like their ancient navigator counterparts. The "journey" of interest here for the systems engineer is an engineering project that is 1) more complex and abstract than geographic travel, 2) has a starting point and destination, 3) with opportunities to become lost or disoriented, 4) and with risks of not reaching the desired destination. The limitations of procedural checklists are well known: a) all the required steps have been performed, b) the checklist boxes are all checked, but c) the result is not acceptable. What is missing is not just some overlooked steps to record, but relational map knowledge that cannot be represented as process steps alone, because is it about a map of something different than process space. This is about the underlying nature of design and exploration of spaces, and not about a certain styles of engineering processes versus others. The history of science, engineering, and mathematics offers evidence that improved cognitive maps of spaces have had profound impact in advancing those fields.

The second article, "Got Phenomena? Science-Based Disciplines for Emerging Systems Challenges," takes on the oft stated pronouncements that systems engineering is not a "real" engineering discipline such as civil. mechanical, chemical, and electrical engineering. The argument is that these fields have "real physical phenomena," "hard science" based laws, and first principles, claiming systems engineering lacks equivalent phenomenological foundations. We counter that the laws and phenomena of traditional disciplines are less fundamental than the system phenomenon from which they spring. This is a reminder of emerging higher disciplines, with phenomena, first principles, and physical laws, with the system phenomenon being the wellspring of engineering opportunities and challenges. Governed by Hamilton's principle, the system phenomenon is a traditional path for derivation of equations of motion or physical laws of so-called "fundamental" physical phenomena of mechanics, electromagnetics, chemistry, and thermodynamics. Examples include ground vehicles, aircraft, marine vessels, biochemical networks, health care, distribution networks, market systems, ecologies, and the Intent of things (IoT).

The third article, "Explicating System Value through First Principles: Re-Uniting Decision Analysis with Systems Engineering," is essential to delivering system value. The systems engineering profession has had a significant focus on improving systems engineering processes. While process plays an important role, the focus on process is often at the expense of foundational engineering axioms and their contribution to system value. Consequently, systems engineers are viewed as process developers and managers versus technical leaders with a deep understanding of how system interactions are linked to stakeholder value. This paper describes how pattern-based systems engineering (PBSE), as outlined within INCOSE's model-based systems engineering (MBSE) initiative, explicates system value through modeling of first principles, re-uniting systems engineering and decision analysis capabilities.

The fourth article, "Innovation, Risk, Agility, and Learning, Viewed as Optimal Control and Estimation," summarizes how optimal control and estimation in "noisy" environments provides a framework to advance understanding of system innovation life cycles and management of decision risks and learning. The ISO15288 process framework and its exposition in the INCOSE *Systems Engineering Handbook* describe system development and other life cycle processes. Concerns about improving the performance of processes in dynamic, uncertain, and changing environments are partly addressed by "agile" systems engineering approaches. Both are typically described in the procedural language of business processes, so it is not always clear whether the different approaches are fundamentally at odds, or just different sides of the same coin. Describing the target system, its environment, and the life cycle management processes using models of dynamical systems allows us to apply earlier technical tools, such as the theory of optimal control in noisy environments, to emerging innovation methods.

The final article is "What Is the Smallest Model of a System?," How we represent systems is fundamental to the history of mathematics, science, and engineering. Model-based engineering methods shift the nature of representation of systems from historical prose forms to explicit data structures more directly comparable to those of science and mathematics. However, using models does not guarantee simpler representation—indeed a typical fear voiced about models is that they may be too complex. Minimality of system representations is of both theoretical and practical interest. The mathematical and scientific interest is that the size of a system's "minimal representation" is one definition of its complexity. The practical engineering interest is that the size and redundancy of engineering specifications challenge the effectiveness of systems engineering processes. How can systems work be made 10:1 simpler to attract a 10:1 larger global community of practitioners?

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.net. 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. For information about sponsoring *INSIGHT*, please contact the INCOSE marketing and communications director at marcom@incose.net. ■

Systems Engineering: The Journal of The International Council on Systems Engineering

Call for Papers

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 manuscript.

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:

https://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.

AUGUST 2024
VOLUME 27/ISSUE 4

Maps or Itineraries? A Systems Engineering Insight from Ancient Navigators

William D. Schindel, schindel@ictt.com

[Editor: This paper for systems engineering foundations refers to the *Systems Engineering Vision 2025* (Copyright 2014 by the International Council on Systems Engineering), INCOSE Systems Engineering Handbook v3.1 (Copyright 2015 by INCOSE), and ISO 15288:2015.]

■ **ABSTRACT**
Processes and procedures are the heart of current descriptions of systems engineering. The "vee diagram," ISO 15288, the INCOSE *Systems Engineering Handbook*, and enterprise-specific business process models focus attention on process and procedure. However, there is a non-procedural way to view systems engineering. This approach is to describe the configuration space "navigated" by systems engineering, and what is meant by system trajectories in that space, traveled during system life cycles. This sounds abstract because we have lacked explicit maps necessary to describe this configuration space. We understand concrete steps of a procedure, so we focus there. But where do these steps take us? And what does "where" mean in this context? Clues are found in recent discoveries about ancient navigation, as well as later development of mathematics and physics. This paper, part I of a case for stronger model-based systems engineering (MBSE) semantics, focuses on the underlying configuration space inherent to systems.

INTRODUCTION

Systems engineering processes. In contemporary discussion of systems engineering, we encounter descriptions of "vees," waterfalls, spirals, and other picturesque metaphors for the work process. In industry or enterprise-specific descriptions (ISO 15288:2015, INCOSE *Systems Engineering Handbook* 2015) of such work processes, the amount of ink and attention devoted to describing process, sequence, or activity usually exceeds by orders of magnitude the amount devoted to describing the information flowing through that process. We ask here why this is the case, and whether there is a more optimum future state for the effective practice of systems engineering. This inquiry is separately extended to include the life cycle trajectory of systems in (Schindel 2015).

MAPS VERSUS ITINERARIES: CONCEPTS OF SPACE

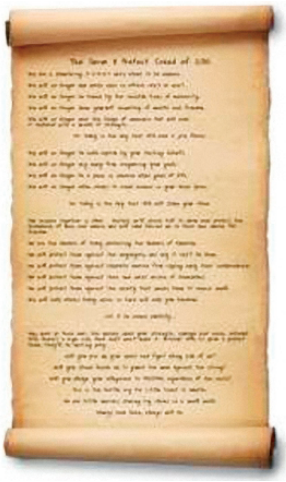
Maps and Itineraries of the Ancient Navigator

In an exhibition at New York University's Institute for the Study of the Ancient World, scholars (Casagrande-Kim et al. 2013) suggested that ancient Greco-Roman navigators did not possess the "ancient maps" of the sort later attributed to them. Instead, it was asserted that these images were generated later, during the Middle Ages, and attributed to the thinking and artifacts of ancient navigators:

> *"Why do we have virtually no ancient maps of the ancient world?" asked a reviewer of the exhibition (Kaylan 2013). "After all, sailors, traders and soldiers had to find their way around. The show's curator, Roberta Casagrande-Kim, distinguishes between a map and an itinerary. The latter 'must have existed aplenty, but being strictly functional probably deteriorated through overuse,' she says. 'A map, however small its focus, suggests a kind of implicit overview, and that is the show's subject.'" (Emphases added)*

In describing how human concepts of space and its representations have evolved, these scholars reported that "Greeks and Romans usually employed what are known as *periploi* ('coastal navigations'), which listed ports and landmarks to facilitate commercial and military sailing, and *itineraria* ('journeys'), lists of locations and distances based on land routes" (Casagrande-Kim et al, 2013) (emphases added).

Figure 1 suggests the conceptual dif-



Itinerary ≠ **Map!**

(*What am I doing?*) (*Where am I?*)

When they eventually did emerge, maps represented a newer idea of the nature of "where."

Figure 1. Map versus itinerary

ference between a map and an itinerary. An itinerary is a sequence of steps whose performance is expected to move us from Point A to Point B. By contrast, a map describes the geographic space of interest, identifying points in geographic space and the relationships between those points. A map is a relational model that answers an infinity of questions that may arise in various situations. A map is not a procedure. By contrast, an itinerary is a stepwise procedure intended for a limited purpose.

A key point examined by scholars is the concept of geographic space held by humans at the time these evolving artifacts were in development (Barkowski 2002). The important notion here is that a map would not emerge sooner than the related cognitive concepts of the space it describes. To appreciate this, we must imagine a time when concepts of geographic space were not yet as developed as today. For example, recall the development of the Mercator cylindrical projection of a sphere (Figure 2), and consider the practical impacts of conceptual challenges that would have preceded its availability.

For purposes of this discussion, the important idea is that people can lack a concept of space that is adequate to what they are trying to do in that space. It is difficult to imagine being without an already familiar concept, but important to understanding the current state of systems engineering. We suggest that equally fundamental concepts are not yet in the regular cognitive maps of the current systems engineer.

Maps and Itineraries of the Systems Engineer

Systems engineering journeys. At least metaphorically speaking, systems engineers must "navigate" a type of "journey," like their ancient navigator counterparts. The "journey" of interest here for the systems engineer is an engineering project:

- More complex and abstract than geographic travel, but …
- it has a starting point and destination,
- with opportunities to become lost or disoriented,
- with risks of not reaching the desired destination.

We will later argue that this is more than just a metaphorical comparison. But first, let us consider the sorts of practical implications at stake for systems engineers.

The limitations of procedural checklists. Experienced practitioners usually admit the following problem situation is a familiar one:

- The junior engineer reports having performed all the required steps
- All the checklist boxes are checked
- But the result is not acceptable.

Why does the junior navigator not recognize, much less avoid, the problem? Often, it is because of deeper knowledge that the senior navigator has internalized through experience, but which is not represented in the official process steps. We will suggest here that what is missing is not just some overlooked steps to record, but relational map knowledge that cannot be represented as process steps alone, because is it about a map of something different than process space.

Are we there yet? Whether the systems engineering journey in a project is based on waterfalls, spirals, or other metaphorical process approaches, certain aspects are inherently iterative, repeating certain activities until a sufficiency is achieved (Figure 3). This is about the underlying nature of design and exploration of spaces, and not about a certain styles of engineering processes versus others.

So, even when individual process steps are clearly defined, a frequently encountered and important question about a systems engineering process is "are we done yet?" This question is answered by different means in different organizations:

- By examining the situation in an underlying information space, or else . . .
- By referring to a checklist of steps that should have been completed, or else . . .
- By referring to schedule or leadership requiring that we be done by now, or …
- By even more arbitrary judgments.

We will argue here that "are we done yet?" should be replaced by "are we there yet?," after we better solidify what "there" and "where" mean.

The above suggest that the practical implications at stake here are significant for the future of systems engineering. The history of science, engineering, and mathematics also offers evidence that improved cognitive maps of spaces have had profound impact in advancing those fields. Two of the most famous cases are the geometrizations offered by Descartes and Hilbert.

The geometrization of algebra. Rene Descartes is credited (Moerdijk 2012) with

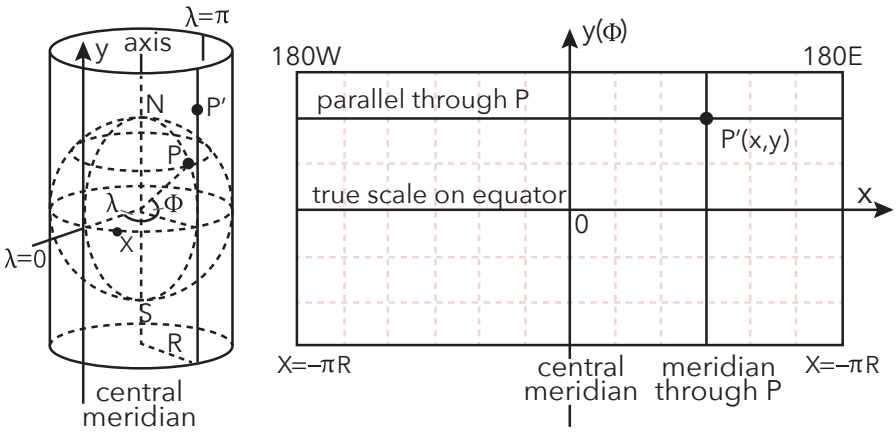


Figure 2. The Mercator projection of sphere onto cylinder

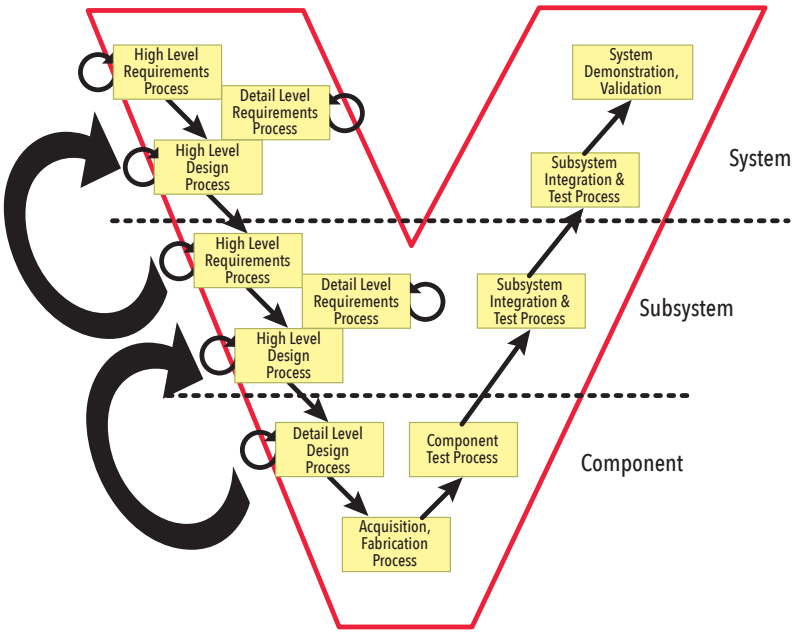


Figure 3. Iteration Is inherent to systems engineering; so when are we done?

moving understanding of symbolic algebra (in particular, algebraic relationships) into a geometric space setting, in which spatial understanding could contribute to understanding of abstract symbolic mathematics, viewed in "Cartesian" coordinates (Figure 4).

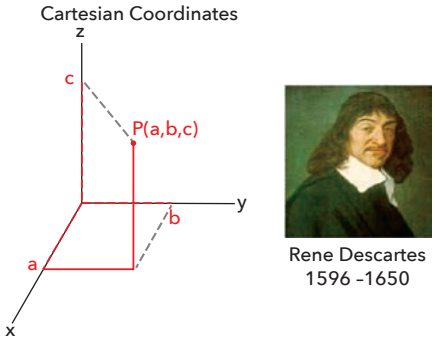


Figure 4. Geometrization of algebra, by Rene Descartes

The geometrization of mathematical functions. As system models also add modeling of (infinite dimensional) behavior, Hilbert Space (Simmons 1963) provided the next required generalization,

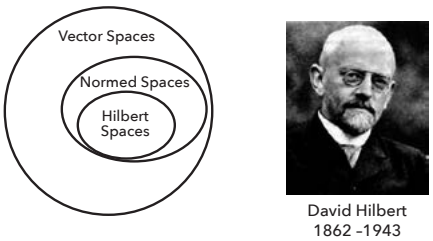


Figure 5. Geometrization of function space, by David Hilbert

supporting a geometrical view of mathematical function (Figure 5). The tools of the modern controls engineer and communications engineer, among others, have been profoundly impacted by geometry-based intuitive basis for more abstract mathematical operations: distance (metric spaces), projections, inner products (including convolutions and frequency transforms). These become applicable to "spatialized" system configuration space, in the MBSE approach described below making what was abstract more concrete and intuitive.

Clues About a Stronger Semantic Model of System Space

It is relatively clear that the description of a sequence of systems engineering process steps (as in ISO/IEC 15288, the INCOSE *Systems Engineering Handbook*, etc.) could be thought of as the metaphorical equivalent of the ancient traveler's itinerary. But, in the same vein, what would be the systems engineering equivalent of the geographic map for such a journey? Through what space is the systems engineer traveling? This is not so immediately clear, but we can begin with what it is not.

A map of the space through which the systems engineer travels:

- is not a list of SE tasks
- is not a model of the SE process — ancient mariners were not traveling through "step space," but "geographic space."

A geographic map describes:

- where we want to end up, along with other points in geographic space where we might conceivably be at a given time
- key relationships between these points, including distance metrics
- expressed in 1, 2, or 3 dimensions: degrees of freedom in geographic space.

So, what is the conceptual systems space through which the systems engineer is navigating? To help answer this, here are a few things that we also know:

- The work of systems engineering produces, and consumes, information
- The space through which the systems engineer navigates would be a map about that information, not the steps of the travel process
- We assert that the space we are interested in should describe the space of possible places for a system of interest to be, good or not, and how they are related to each other: the configuration space of the system
- We know one kind of map about information: an information model (for example, an entity-relationship or similar model)
- The hard sciences provide, in the maps for physics, chemistry, thermodynamics, and other domains, representations of underlying relationships (laws)
  - Frequently represented in the form of mathematical equations.
  - These relationships and their impact on systems space are the focus of attention: Imagine instead trying to learn chemistry by studying the process of cooking!
- Can systems science provide maps in the form of underlying systemic relationships?

Semantic models. INCOSE MBSE thought leadership has called for "stronger semantic models" (Long 2014a and 2014b) to support the future progress of model-based systems engineering. This refers to the notion that, while current and historical modelling language and data exchange standards provide "metamodel" underpinnings, additional progress is needed.

We strongly agree with the call for stronger underlying MBSE semantics. Before discussing that subject, we recall what is meant here by "semantics".

There is an unfortunate practice in popular culture to use the term "semantics" as a dismissive pejorative, as if that term meant "insignificant detail" or "hair-splitting." To the contrary, "semantics" defines fundamental meaning, whether referring to formal engineering models, databases, cognition, or everyday natural language. Nothing could be more important to the success of human endeavor than shared semantics (meaning) that is sufficient for the activities in which humans engage. For

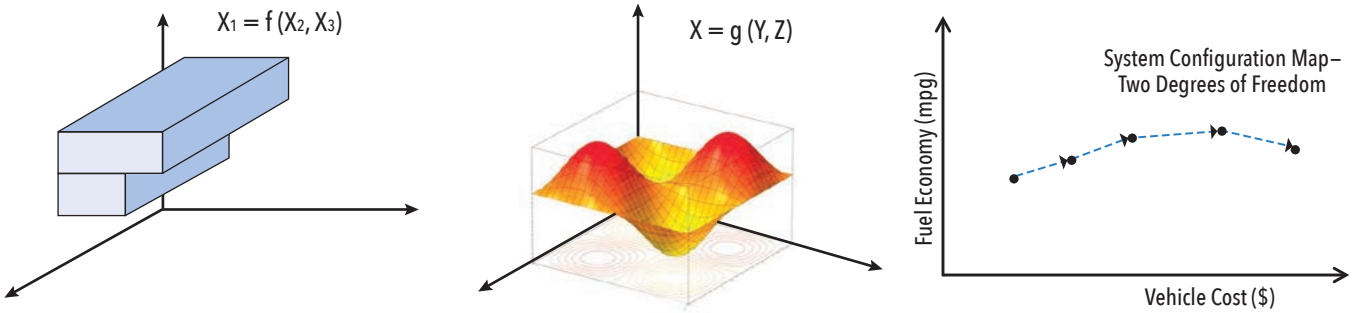


Figure 6. System configuration (degrees of freedom) space, constrained by (discrete and continuous) modeled relationships representing semantics, laws, designs

Figure 7. A simple sub-space of configuration space

purposes of this paper, we define "semantics" of a conceptual space as the degrees of freedom of that space, and the relationships between them—the "map" of the space.

An example of "semantics" in the technical space of science, engineering, and mathematics is Newton's second law, sometimes expressed in equation form: $F=mA$. In discovering this natural law, Newton not only arrived at a quantitative relationship, but also a stronger (and inherently circular) definition of the concepts (mass, force, acceleration) that it relates. This was not just a matter of refining dictionary definitions, but a fundamental recasting of the relational cognitive map of the natural world, with profound practical consequences. (The same was true for those who followed Newton, refining that map.)

These three things are inter-related:

- System configuration space—the space described by the degrees of freedom of conceivable systems, in which each point represents one system configuration (Figure 6)
- Relational models, constraining those same degrees of freedom with respect to each other, often mathematical or other relational models (including various types of information models)
- Semantic "meaning" expressed in the form of relationships

Figure 6, representing a subspace of system configuration space, is not the same as the equations, words, or model views (for example, SysML) that might be used to describe the set of instance points within it. This is an important reminder that a view of a model is not a direct view of the configuration space it describes, but instead a compressed representation of constraints that define such a configuration space—just as Descartes noted that viewing an algebraic equation is not the same as viewing the geometric space it describes—and both have their place. (Note that system models describe both discrete and continuous degrees of freedom, as shown in Figure 6.)

What we usually refer to as "modelling languages" (for example, mathematical languages, database modelling languages, systems modelling languages) are not themselves the semantics of the spaces they will be used to describe. The description of English as a language does not itself describe the struggles of Hamlet that Shakespeare encoded using English.

However, we know that architectural patterns, expressed in those modelling languages, can be used to describe the semantics of train systems or manufacturing processes. That is, the semantics of a lower-level language can be used to encode the semantics of a higher level "language," formalizing the latter (Schindel 2011b). Semantic models of systems engineering occur at different levels of abstraction. The following example list proceeds from more specific to more abstract cases:

1. Model of a specific automobile instance, configured as sought by its owner
   - Example of use: Represents whether cruise control option is equipped
2. Model of a product line of automobiles, optimized by designers and planners (ISO26550 2013)
   - Example of use: Defines which automobile models allow cruise control option
3. Architectural framework model (ISO 42010 2011) of consumer automobiles, shared across suppliers active in the automotive domain
   - Example of use: Defines semantics, behavior of "cruise control feature"
4. Metamodel of a specific system modelling language, semantically capable of expressing concepts appropriate to its intended use, along with syntax and views specific to that language
   - Example of use: Defines how stakeholder features will appear in model views
5. Metamodel of concepts sufficient for the purposes of systems engineering or science, independent of the modelling languages that will express them in specific cases.
   - Example of use: Defines the semantics of "stakeholder feature"

The entire configuration "system DNA" of a given system configuration or series of life cycle configurations can practically be captured by properly configured modelling, product lifecycle management (PLM), or other tools, as further illustrated in Schindel, Lewis, Sherey, and Sanyal (2015).

The dimensionality of this configuration space is high, so we don't typically view the whole space at one time, preferring instead to view sub-spaces. Figure 7 is a simple example.

The constraints that result in the curve of Figure 7 remind us that a further compression of configuration instance information is provided by modelled relationships:

- Mathematical equations (couplings, dependencies)
- Information models (E-R, SysML, IDEF, etc.)
- Requirements statements, viewed as transfer functions (Schindel 2005b).

Moreover, pattern-based systems engineering (PBSE) methods permit even further compression of these views (Schindel 2011b)—layers of compression are likewise possible. Most of the sub-space relationships are not linear, so certain ideas such as linear combinations and frequency domain transfer functions won't apply in the linear sense. However, other geometric aspects, such as distance norms and projections, do still apply. Of course, we'd likely add many more degrees of freedom (weight, range, etc.)—so system maps will tend to be high dimension, and subject to "slicing" into multiple views. During innovation / development cycles, and some life cycles, the "current configuration" may involve sets of ranges or lists, instead of individual points, so the trajectory becomes an ordered series of envelopes.

MOVING TO A STRONGER SEMANTIC MODEL OF SYSTEM CONFIGURATION SPACE

What are the degrees of freedom (relatable variables) needed by system models to describe system space? Do system modeling languages (SysML, OPM, IDEF, etc.) answer this? Some thought

What Is the Smallest Model of a System?

William D. Schindel
ICTT System Sciences
schindel@ictt.com

Abstract. How we represent systems is fundamental to the history of mathematics, science and engineering. Model-based engineering methods shift the nature of representation of systems from historical prose forms to explicit data structures more directly compatable to those of science and mathematics. However, using models does not guarantee simpler representation—indeed a typical fear voiced about models is that they may be too complex.

Minimality of system representations is of both theoretical and practical interest. The mathematical and scientific innterest is that the size of a system's "minimal representation" is one definition of its complexity. The practical engineering interest is that the size and redundancy of engineering specifications challenge the effectiveness of systems engineering processes. INCOSE thought leaders have asked how systems work can be made 10:1 simpler to attract a 10:1 larger global community of practitioners. And so, we ask: What is the smallest model of a system?

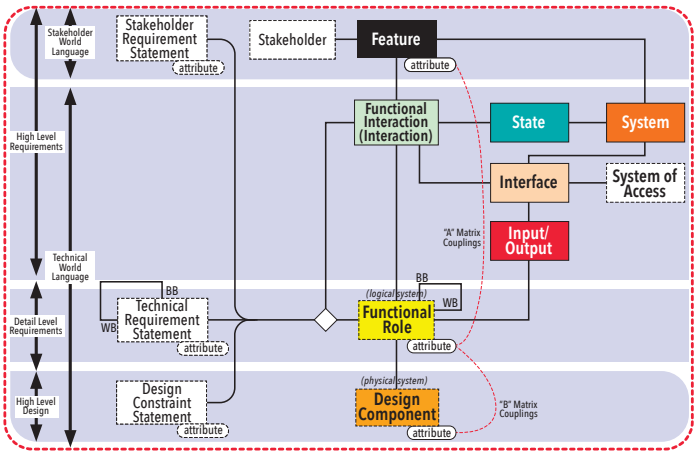


Figure 8. What Is the smallest model of a system?

leaders (Long 2014a and 2014b) agree that such languages are more syntactical or view-oriented than about underlying semantics, with none of them currently providing a complete semantic model of the systems they describe. Based on the above arguments, it is perhaps too much to expect that they should, because they are intended to provide views into such an underlying system configuration space. Nevertheless, many of the ideas described in these modeling languages and other frameworks (for example, OMG 2012 and ISO 10303 U'Ren 2003) do cover a significant part of the territory. Along with a language description, modeling language specifications typically include an effort to describe the underlying system configuration space (even if entangled a bit in the description of the modeling language), for lack of a pre-existing community agreement on that underlying space.

In the spirit of the physical sciences, we therefore have asked "What is the smallest model of a system?" for effective descriptions in the work of engineering and science, and independent of any specific modeling language. Pursued over a number of years and tests, this work showed that contemporary system models are often both semantically too big (redundant) and too small (missing important information), at the same time (Schindel 2011b).

In our practice with others across multiple system domains (Schindel and Smith 2002, Bradley et al. 2010, Schindel 2012b, Berg 2014), this led over several decades to a formal model of the semantics of the underlying system space, referred to as the S*Metamodel. Figure 8 illustrates a key subset summary of the longer formal S*Metamodel specification (ICTT 2009 and 2013).

Formal mappings (profiles) of the S*Metamodel have been created for a number of existing third-party COTS modeling tools, engineering databases, PLM systems, and standards-based language offerings. These increase the power of the existing industry assets by strengthening their expressive power and semantic compatibility, in comparison to simple data exchange interfaces (Schindel, Lewis, Sherey, Sanyal 2015). These systems and their users are enabled to represent and understand systems in S*Space.

Further Evidence of the Need

Why is such a transition in thought and practice important? An ancient navigator would not have been in a position to articulate the need for a map in the same terms we would use today, so today's systems navigators may face the same kind of barriers to visions of the future.

Further evidence is here offered in three areas:

1. **System interactions**: One reference is the history of improvement of human life during the last three hundred years, driven by the fruits of science and engineering as they explicated and harvested deeper understanding of nature. A prime connection of systems and that history is the central role of physical interactions as the basis of all scientific laws in the physical sciences, discovered, expressed, and exploited over those three centuries to improve human life. We assert that physical interactions between parts are likewise the foundational perspective of the science and engineering of systems. Interactions accordingly play a central part in the S*Metamodel (Schindel 2013a). However, these interactions are not necessarily recognized in the same way by contemporary system modeling languages and tools or are in other cases merely tolerated by them.
2. **System failures**: Human engineered systems have purpose, at the risk of failure in that purpose. Analysis of failure modes and effects (FMEA, FMECA, etc.) and other forms of risk analysis are central to systems engineering and are likewise fundamental to the S*Space described by the S*Metamodel (Schindel 2010). Purpose is not an add-on, and neither is failure in that purpose.
3. **System requirements**: Systems engineers know that requirements are important, but they are most frequently conceived as the prose statements used to represent them to humans. Efforts by the suppliers of engineering tools and databases have brought forth databases and later models that incorporate and link to and among these textual structures. However, these text representations are the "prose equations" of the non-linear extension of transfer functions (Schindel 2005), even if not recognized as such. Imagine an engineering world in which mathematical equations were viewed as being primarily the strings of text that represent them. Accordingly, the related transfer function abstraction is fundamental to the S*Metamodel's integration of requirements.

Information vs. Process: Re-Integrating Systems Engineering Maps and Itineraries

Once a stronger semantic model of system space is in hand, its re-integration with systems processes and procedures is possible. We have found this has good positive impact on the traditional procedures with which we re-integrate that systems space, making those processes and procedures more effective while respecting their historical roots and values.

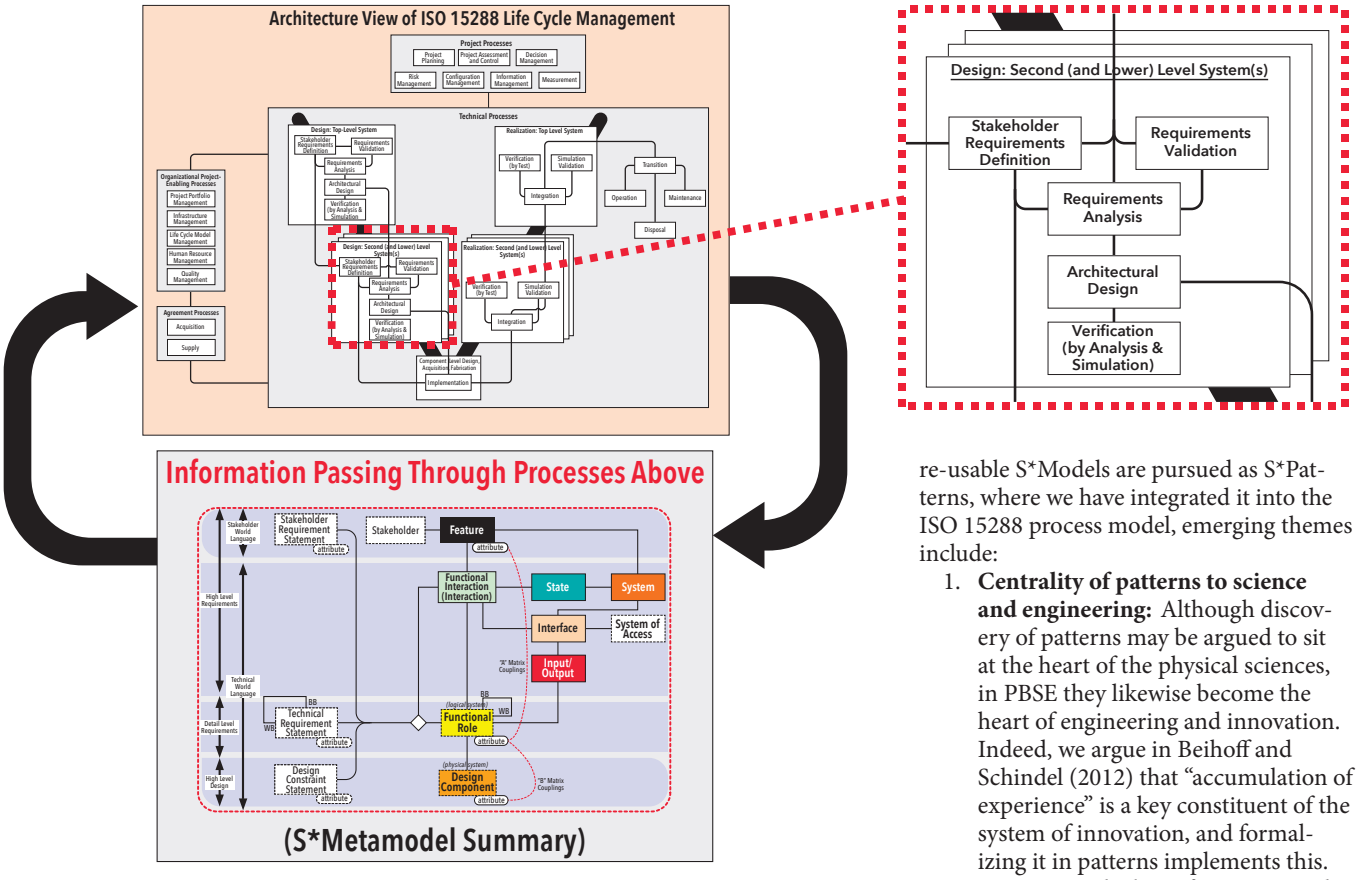


Figure 9. Process versus information

For example, we have created formal models of the ISO15288 processes, integrating with that well-known framework while giving new insight and power to its implementation. Figure 9 summarizes the notion that this paper began with: the systems engineering process (summarized at the top of Figure 9 by ISO 15288 process areas) consumes and produces information. By using a stronger semantic model of that information (S*Metamodel summarized at the bottom of Figure 9), we strengthened each of the systems engineering processes that consume and produce that information.

A part of that strengthening was to introduce into those systems engineering process models not only the option for MBSE models of target systems and their views, but the further notion that these models can be constructed from model-based S*Patterns. This is discussed in the next section.

Trajectories, Persistent Memories Patterns: Roads Already Travelled

System configuration trajectories (Figure 9 lower right) are not just important during development of a single system generation. Across the life cycles of multiple systems, we have the splitting evolution of systems that emerge as responses to their environments. What is the configuration space for these evolving systems across multiple family life cycles (Figure 10)?

The same underlying S*Metamodel, along with the "System of Innovation Pattern" (Beihoff and Schindel 2012 and Schindel 2013b) supports all these, including more specialized system family, product line, or architectural patterns and frameworks (Fig. 11). In addition to our own firm's work in pattern-based systems engineering (PBSE) over several decades, PBSE based on these S*Patterns is also being pursued and practiced by the Patterns Challenge Team of the INCOSE/OMG MBSE initiative (INCOSE Patterns Team 2014), and the subject of several related IS2015 papers (Cook and Schindel 2014; Nolan, Pickard, Russell, and Schindel 2015; Peterson and Schindel 2015; Schindel, Lewis, Sherey, and Sanyal 2015).

When persistent memory of configurable re-usable S*Models are pursued as S*Patterns, where we have integrated it into the ISO 15288 process model, emerging themes include:

1. **Centrality of patterns to science and engineering:** Although discovery of patterns may be argued to sit at the heart of the physical sciences, in PBSE they likewise become the heart of engineering and innovation. Indeed, we argue in Beihoff and Schindel (2012) that "accumulation of experience" is a key constituent of the system of innovation, and formalizing it in patterns implements this. Patterns, as the basis for engineered platforms and product lines, become the equivalent of theoretical frameworks and paradigms in science.
2. **Intellectual assets:** After several decades of investment in computer software, the Financial Accounting Standards Board (FASB) formally recognized accounting for that investment on a capitalized asset basis, joining "bricks and mortar" as a financial asset. Since that time, annual U.S. investment in intangible assets has grown to exceed investment in tangibles. The model-based economy is arriving. S*Patterns satisfy the criteria of being a form of software, eligible for that capitalization of investment in systems IP (Schindel 2007 and Sherey 2006)
3. **Process patterns:** The systems engineering process, or the larger innovation process, are themselves

Figure 10. Evolving systems, over multiple life cycles

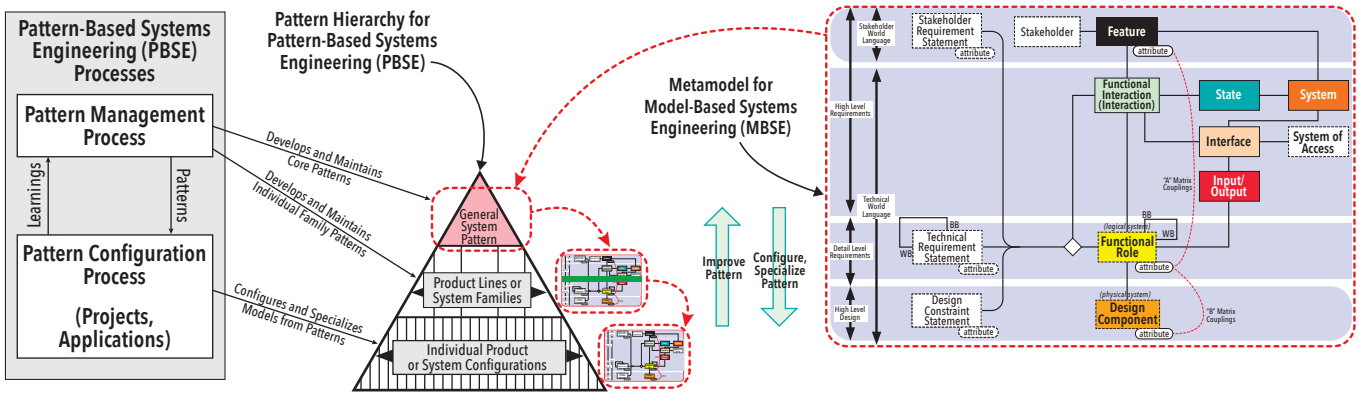


Figure 11: Evolving families of systems, pattern-based systems engineering (PBSE)

systems, and may be modeled as such (Beihoff and Schindel 2012; Schindel 2013b; Schindel, Ahmed, Hanson, Peffers, Kline 2011). Accordingly, there are also S*Pattern representations for these systems, as we have created for ISO 15288.

Beginning at the INCOSE IW2015 MBSE workshop, we will examine the agile systems representation in this model-based framework (Dove and Schindel 2015).

CONCLUSIONS, IMPLICATIONS AND FUTURE WORK

1. We assert, and have offered argument and evidence above, that the MBSE model-centric vision expressed by INCOSE *Vision 2025* will require progress in shared understanding of the underlying semantic model of system space, and that this will be needed independent of specific modelling language/modelling view semantics, even when they are themselves standards-based. Indeed, these languages and systems can themselves build upon and gain from such a shared underlying semantic model.
2. Current procedure-based systems engineering and innovation processes can be made more effective by increasing the focus on underlying information vs. procedure, without abandoning the value of procedural foundations, and with these impacts:
   - Knowing "where you are, not just what you are doing"
   - Simplification, while speeding and improving outcomes
   - Improved ability to understand, think critically about, represent, and communicate
   - "the current situations" in projects, coupled with more effective risk management (Schindel 2011c)
   - Increased agility of the overall System of Innovation (Dove and LaBarge 2014)
   - Availability of an MBSE model of ISO 15288, incorporating PBSE options
   - Improved capabilities for even the currently available generation of automated aids, modelling tools, and PLM systems
   - Realizing more of INCOSE *Vision 2025*. ■

REFERENCES

- Barkowsky, Thomas 2002. *Mental Representation and Processing of Geographic Knowledge*. Berlin, DE: Springer.
- Berg, E. 2014. "Affordable Systems Engineering: An Application of Model-Based System Patterns to Consumer Packaged Goods Products, Manufacturing, and Distribution." Presented at the Annual International Workshop of INCOSE (MBSE Workshop), Los Angeles, US-CA, 25-28 January.
- Beihoff, B., and W. Schindel. 2012. "Systems of Innovation I: Models of Their Health and Pathologies." Paper presented at the 22nd Annual International Symposium of INCOSE, Rome, IT, 9-12 July.
- Bradley, J., M. Hughes, and W. Schindel, 2010. "Optimizing Delivery of Global Pharmaceutical Packaging Solutions, Using Systems Engineering Patterns." Paper presented at the 20th Annual International Symposium of INCOSE, Chicago, US-IL, 12-15 July.
- Casagrande-Kim, Roberta, et al. n.d. NYU ISAW web site and bibliography on ancient cartography: http://isaw.nyu.edu/exhibitions/space/bibliography.html.
- Cook, D., and W. Schindel. 2015. "Utilizing MBSE Patterns to Accelerate System Verification." Paper presented at the 25th Annual International Symposium of INCOSE, Seattle, US-WA, 13-16 July.
- Dove, R., and R. LaBarge. 2014. "Fundamentals of Agile Systems Engineering—Part 1" and "Part 2." Papers presented at the 24th Annual International Symposium of INCOSE, Las Vegas, US-NV, 30 June – 3 July.
- Dove, R., and W. Schindel. 2015. "Agile Modeling and Modeling Agile Systems." Presented at the Annual International Workshop of INCOSE, Los Angeles, US-CA, 24-27 January.
- Estafan, J. 2008. "Survey of Model-Based Systems Engineering (MBSE) Methodologies." INCOSE MBSE Initiative.
- ICTT. 2009. "Systematica Metamodel" Version 7.1. Methodology Release 4.0. ICTT System Sciences, 29 May.
- ICTT. 2013. "Abbreviated Systematica 4.0 Glossary" P3125 Ver. 4.2.2. ICTT System Sciences.
- Moerdijk, Ieke. 2012. "Descartes and the Geometrization of Algebra." Descartes-Huygens Lecture, Radboud University, Nijmegen, NL, 3 April.
- INCOSE Handbook. 2015. *Systems Engineering Handbook: A Guide for System Life Cycle Processes and Activities* Version 4. Edited by D. Walden, G. J. Roedler, K. J. Forsberg, R. D. Hamelin, and T. M. Shortell. International Council on Systems Engineering, San Diego, US-CA: Wiley.
- INCOSE Vision 2025. 2014. "A World in Motion: Systems Engineering Vision 2025." International Council on Systems Engineering, San Diego, US-CA.
- INCOSE Patterns Team. 2014. INCOSE/OMG MBSE Initiative: Patterns Challenge Team 2013-2014. http://www.omgwiki.org/MBSE/doku.php?id=mbse:patterns:patterns.

- ISO 15288. 2015. ISO/IEC/IEEE 15288. Systems Engineering—System Life Cycle Processes. International Organization for Standardization, Geneva, CH: ISO.
- ISO 26550. 2013. ISO/IEC 26550. Systems and Software Engineering—Reference Model for Product Line Engineering and Management. International Organization for Standardization, Geneva, CH: ISO.
- ISO 42010. 2011. ISO/IEC/IEEE 42010. Systems and Software Engineering—Architecture Description. International Organization for Standardization, Geneva, CH: ISO.
- Long, David 2014a. "Model-Based Systems Engineering at the Age of Eight." Presented at NDIA GVSETS Conference, Troy, US-MI, August.
- Long, David, 2014b. Keynote Address to INCOSE Great Lakes Regional Conference on Systems Engineering, Schaumburg, US-IL, 10 October.
- Kaylan, Melik. 2013. "A World Without Maps." *The Wall Street Journal* 10.29-30.2013.
- Mercator. 2014. "The Mercator Projection." Wikipedia: http://en.wikipedia.org/wiki/Mercator_projection.
- Nolan, A., A. Pickard, J. Russell, and W. Schindel. 2015. "When Two is Good Company, but More is Not a Crowd." Paper to be presented at the 25th Annual International Symposium of INCOSE, Seattle, US-WA, 13-16 July.
- OMG. 2012. "OMG Systems Modeling Language" Version 1.3. Object Management Group, June.
- Peterson, T., and W. Schindel. 2015. "Autonomous Ground Vehicle Platforms and Model-Based System Patterns: An Example." Paper presented at the 25th Annual International Symposium of INCOSE, Seattle, US-WA, 13-16 July.
- Schindel, W. 2005a. "Pattern-Based Systems Engineering: An Extension of Model-Based SE." Tutorial presented at the 15th Annual International Symposium of INCOSE, Rochester US-NY, 13-16 July.
- Schindel, W. 2005b. "Requirements Statements are Transfer Functions: An Insight from Model-Based Systems Engineering." Paper presented at the 15th Annual International Symposium of INCOSE, Rochester US-NY, 13-16 July.
- Schindel, W. 2007. "Are Patterns Software?" ICTT System Sciences, January.
- Schindel, W. 2010. "Failure Analysis: Insights from Model-Based Systems Engineering." Paper presented at the 20th Annual International Symposium of INCOSE, Chicago, US-IL, 12-15 July.
- Schindel, W. 2011b. "What Is the Smallest Model of a System?" Paper presented at the 21st Annual International Symposium of INCOSE, Denver, US-CO, 20-23 June.
- Schindel, W. 2011c. "The Impact of 'Dark Patterns' on Uncertainty: Enhancing Adaptability In The Systems World." Presented at the INCOSE Great Lakes 2011 Regional Conference on Systems Engineering, Dearborn, US-MI.
- Schindel, W. 2012a. "Introduction to Pattern-Based Systems Engineering (PBSE)." Presented at the INCOSE Finger Lakes Chapter Webinar, 26 April.
- Schindel, W. 2012b. "Integrating Materials, Process, & Product Portfolios: Lessons from Pattern-Based Systems Engineering." in *Proc. of Society for Advancement of Materials and Process Engineering* (SAMPE).
- Schindel, W. 2013a. "Interactions: At the Heart of Systems." Presented at the INCOSE Great Lakes Regional Conference on Systems Engineering, West Lafayette, US-IN, October.
- Schindel, W. 2013b. "Systems of Innovation II: The Emergence of Purpose." Paper presented at the 23rd Annual International Symposium of INCOSE, Philadelphia, US-PA, 24-27 June.
- Schindel, W. 2014. "The Difference Between Whole-System Patterns and Component Patterns: Managing Platforms and Domain Systems Using PBSE." Presented at the INCOSE Great Lakes Regional Conference on Systems Engineering, Schaumburg, US-IL, October.
- Schindel, W. 2015. "System Life Cycle Trajectories: Tracking Innovation Paths Using System DNA." Paper presented at the 25th Annual International Symposium of INCOSE, Seattle, US-WA, 13-16 July.
- Schindel, W., S. Lewis, J. Sherey, and S. Sanyal. 2015. "Accelerating MBSE Impacts Across the Enterprise: Model-Based S*Patterns." Paper presented at the 25th Annual International Symposium of INCOSE, Seattle, US-WA, 13-16 July.
- Schindel, W., S. Peffers, J. Hanson, J. Ahmed, and W. Kline. 2011. "All Innovation is Innovation of Systems: An Integrated 3-D Model of Innovation Competencies." *Proc. of ASEE 2011 Conference*, American Association for Engineering Education.
- Schindel, W., and T. Peterson. 2013. "Introduction to Pattern-Based Systems Engineering (PBSE): Leveraging MBSE Techniques." Tutorial at the 23rd Annual International Symposium of INCOSE, Philadelphia, US-PA, 24-27 June.
- Schindel, W., and V. Smith 2002. "Results of applying a families-of-systems approach to systems engineering of product line families." SAE International, Technical Report 2002-01-3086.
- Sherey, J. 2006. "Capitalizing on Systems Engineering." Paper presented at the 16th Annual International Symposium of INCOSE, Orlando, US-FL, 9-13 July.
- Simmons, George F. 1963. *Introduction to Topology and Modern Analysis*, Chapter 10: Hilbert Spaces. McGraw-Hill.
- U'Ren, J. 2003. "An Overview of AP233: STEP's Systems Engineering Standard.", ISO 10303 AP233 Working Group, 20 October.

ABOUT THE AUTHOR

[Editor: Author biography was current when the paper was published in 2015.]

Bill Schindel is president of ICTT System Sciences (www.ictt.com), a systems engineering company. His 40-year engineering career began in mil/aero systems with IBM Federal Systems, Owego, NY, included service as a faculty member of Rose-Hulman Institute of Technology, and founding of three commercial systems-based enterprises. He has led and consulted on improvement of engineering processes within automotive, medical/health care, manufacturing, telecommunications, aerospace, and consumer products businesses. Bill has led the development and practice of Systematica™ methodology for pattern-based systems engineering. He earned the BS and MS in mathematics. At the 2005 INCOSE International Symposium, he was recognized as the author of the outstanding paper on modeling and tools, co-led a 2013 research project on the science of systems of innovation within the INCOSE System Science Working Group, and currently co-leads the patterns challenge team of the OMG/INCOSE MBSE initiative. Bill is an INCOSE CSEP, and president of the Crossroads of America INCOSE chapter.

Got Phenomena? Science-Based Disciplines for Emerging Systems Challenges

Bill Schindel. schindel@ictt.com

[Editor: This paper for systems engineering foundations refers to the *Systems Engineering Vision 2025* (Copyright 2014 by the International Council on Systems Engineering).]

■ **ABSTRACT**

Engineering disciplines (civil, mechanical, chemical, electrical) sometimes argue their fields have "real physical phenomena", "hard science" based laws, and first principles, claiming systems engineering lacks equivalent phenomenological foundation. We argue the opposite, and how replanting systems engineering in model-based systems engineering (MBSE)/pattern-based systems engineering (PBSE) supports emergence of new hard sciences and phenomena-based domain disciplines.

Supporting this perspective is the system phenomenon, wellspring of engineering opportunities and challenges. Governed by Hamilton's principle, it is a traditional path for derivation of equations of motion or physical laws of so-called "fundamental" physical phenomena of mechanics, electromagnetics, chemistry, and thermodynamics.

We argue that laws and phenomena of traditional disciplines are less fundamental than the system phenomenon from which they spring. This is a practical reminder of emerging higher disciplines, with phenomena, first principles, and physical laws. Contemporary examples include ground vehicles, aircraft, marine vessels, and biochemical networks; ahead are health care, distribution networks, market systems, ecologies, and the IoT.

INTRODUCTION

As a formal body of knowledge and practice, systems engineering is much younger than the more established engineering disciplines, such as civil, mechanical, chemical, and electrical engineering. Comparing their underlying scientific foundations to some equivalent in systems engineering sometimes arises as a dispute, concerning whose profession is "real" engineering based on (or at least later explained by) hard science, with tangible physical phenomena, and accompanied by physical laws and first principles. This paper argues for a different perspective altogether (Figure 1), and the reader exploring this paper is warned to avoid the trap of the seemingly familiar in parsing the message.

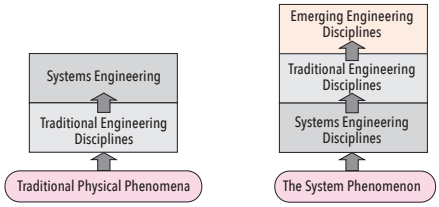


Figure 1. En garde! Not what you may be expecting

Beyond that argument, this paper addresses a more pragmatic goal—the means of identifying and representing the tangible physical phenomena that emerge in new system domains, along with their respective physical laws and first principles. This is of more than philosophical or professional significance. Challenged by numerous issues in emerging systems, society has an interest in organizing successful approaches to the scientific understanding of laws and first principles about, and engineering harnessing of, the related phenomena. Individuals entering or navigating the technical professions likewise have personal interests in this evolving roadmap.

While recognizing the formidable works of systems theorists in these still early days of systems engineering (Ashby 1956, Bertalanffy 1969, Braha et al. 2006, Cowan

et al. 1994, Holland 1998, Prigogine 1980, Warfield 2006, Wymore 1993), this paper focuses on even earlier contributions of science and mathematics to the flowering of engineering's impact over the last three centuries. We will extract the "system phenomenon" at the center of that foundation and consider its impacts and implications for systems engineering practice. This perspective helps us understand the phase change that systems engineering is going through, as model-based representations enable the framework that has already had profound impact in the traditional science/engineering paired disciplines.

Section 2 of this paper reminds us of the "phase change" that occurred in science, technology, engineering, and mathematics (STEM) approximately 300 years earlier, when means of representation advanced, and argues efficacy from the pragmatic perspective of the dramatic impacts on human life. Section 3 argues that we are now in the early days (when trends can still be confusing) of a similar phase change in the STEM of general systems. Section 4 provides the main argument, introduces the system phenomenon, and asserts that it is not only the hard physical phenomena basis for systems engineering, but surprisingly also for all the traditional disciplines' phenomena, reversing the "who's got real phenomena?" argument. This section also suggests the means of identifying and representing the tangible physical phenomena emergent at all levels, and their respective physical laws and first principles. Section 5 returns to the subject of current trends in systems engineering, the need to strengthen its foundation, and the opportunity to use model representation of the system phenomenon to that end. Section 6 concludes with implications for action.

PHASE CHANGE EVIDENCE: EFFICACY OF HARD SCIENCE, PHENOMENA-BASED, STEM DISCIPLINES

Science, Technology, Engineering, and Mathematics —300 Years of Impact

Our pragmatic argument is based on assessing the impact of the physical sciences and mathematics on engineering by their joint efficacy in improving the human condition. In a matter of 300 years (from around Newton), the accelerating emergence of STEM has lifted the possibility, quality, and length of life for a large portion of humanity, while dramatically increasing human future potential (Mokyr 2009, Morris 2012, Rogers 2003). Among the measures of this impact are Figures 2, 3, and 4. By the close of the twentieth century, the learning and impacts of STEM along with other factors (for example, market capitalism as a driver of prosperity, as in

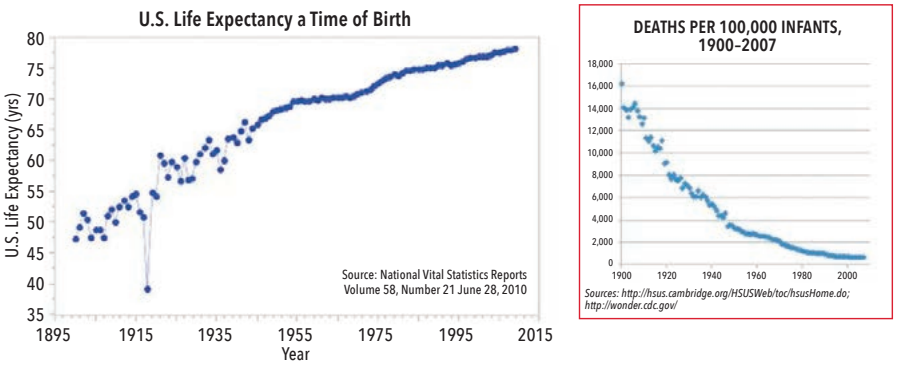


Figure 2. The length of human life has been dramatically extended

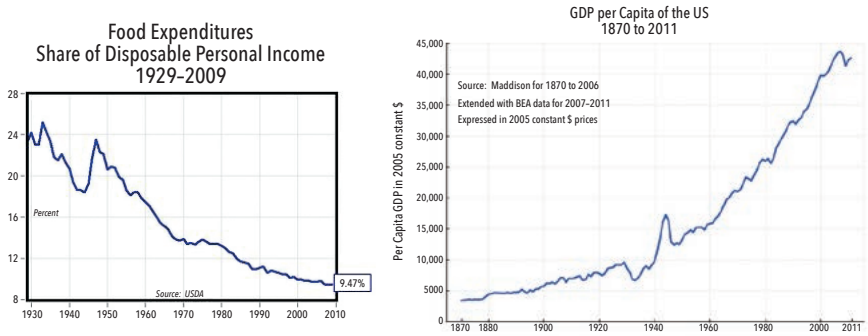


Figure 3. Simply feeding ourselves consumes less labor and time

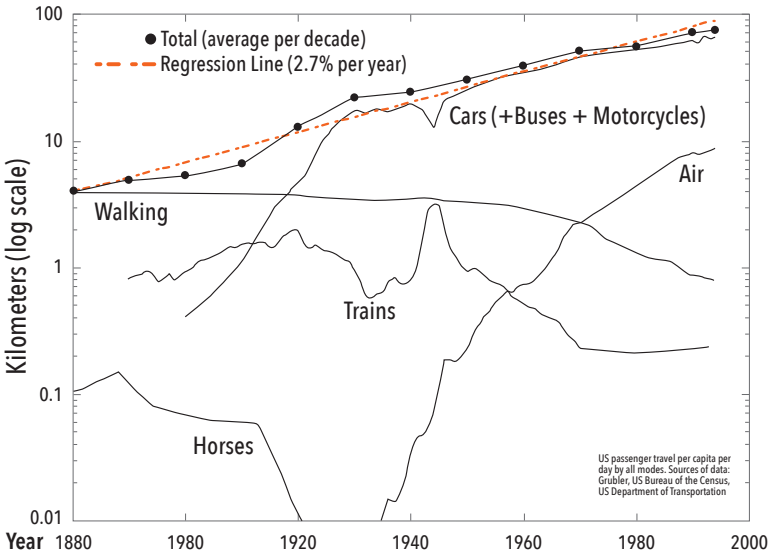


Figure 4. The range of individual human travel has vastly extended

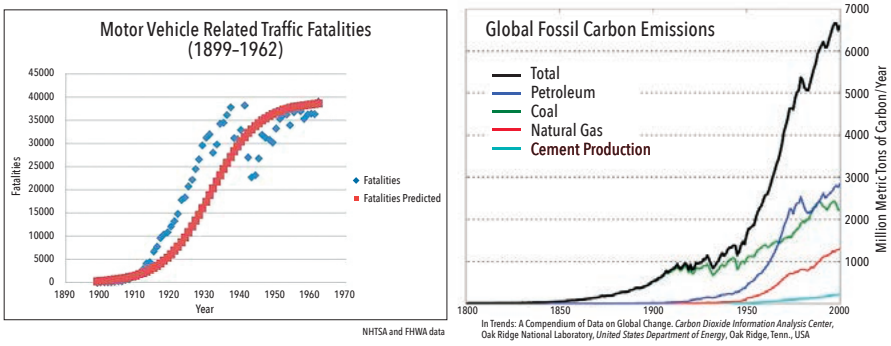


Figure 5. More available energy and mobility have brought unintended consequences

Friedman (1980)) were increasingly recognized as critical to individual and collective human prosperity.

During that same period, the human-populated world has become vastly more interconnected, complex, and challenging. New opportunities and threats have emerged, in part out of less positive impacts of human applications of STEM. Understanding and harnessing the possibilities have become even more important than before, from the smallest known constituents of matter and life, to the largest scale complexities of networks, economies, the natural environment, and living systems. Figure 5 illustrates other parameters of these impacts.

Because we argue here from the efficacy shift as STEM advanced, one might question how much other causes (for example, market capitalism as noted above) accounted for these advances. To remember that these shifts were more than just correlations in time, Table 1 reminds us of some of the more familiar and yet dramatic STEM-based advances associated with the above impacts:

Table 1. STEM drivers that contributed to the above impacts

Impact	Notable STEM Drivers (sample only)
Increased life expectancy	Life sciences, nutritional science
Reduced infant mortality	
Reduced cost of food production	Agronomy, herbicides, fertilizers, mechanization
Increased GDP per capita	Mechanized production, mechanized distribution
Increased range of travel	Vehicular, civil, and aerospace engineering
Increased traffic fatalities	Vehicular engineering, civil engineering
Increased carbon emissions	Vehicular engineering; mechanized production

Table 2. Phenomenon-based disciplines

Engineering Discipline	Phenomena	Scientific Foundations	Representative Scientific Laws
Mechanical Engineering	Mechanical Phenomena	Physics, Mechanics, Mathematics, . . .	Newton's Laws, others
Chemical Engineering	Chemical Phenomena	Chemistry, Mathematics, . . .	Periodic Table, others
Electrical Engineering	Electromagnetic Phenomena	Electromagnetic Theory	Maxwell's Equations, others
Civil Engineering	Structural Phenomena	Materials Science, . . .	Hooke's Law, others

"Phase Changes": Emergence of Science and Engineering as Phenomena-Based Disciplines

Over those three centuries, the "hard sciences", along with the engineering disciplines and technologies based on those sciences, are credited with much of this amazing societal progress, as well as some related challenges (Mokyr 2009, Morris 2012, Rogers 2003). Our point here is the enormous impact of these "traditional" (at least, over 300 short years) disciplines, as their foundations emerged in understanding of physical phenomena and related predictive and explanatory models.

How can the foundational roots of systems engineering be compared to engineering disciplines already seen as based on the "hard sciences?" As illustrated in Table 2, the traditional engineering disciplines have their technical bases and quantitative foundations in what emerged as physical sciences about what came to be understood as physical phenomena.

It wasn't always this way, as seen from the shift that began to occur just three centuries ago. It is informative to remember the "phase changes" that occurred in what are now considered the traditional disciplines, by recalling the history of physics before Newton, chemistry before Lavoisier & Mendeleev, and electrical science before Faraday, Hertz, and Maxwell, versus what followed for each (Cardwell 1971, Forbes et al. 2014, Pauling 1960, Servos 1996, Westfall 1980). All of these domains had earlier, less effective, bodies of thought, generated by those attempting to answer questions and, in some cases, provide practical benefits. Instead of dismissing alchemy, astrology, pre-Copernican cosmology, and their counterparts, we can instead see them as grappling with phenomena without the benefit of sufficiently powerful mathematics and the verification mechanisms of experiment and refutation to test against reality what we would now call models.

SYSTEMS ENGINEERING IS STILL YOUNG

Contemporary specialists in individual engineering disciplines (for example, civil, mechanical, chemical, electrical) sometimes argue that their fields are based on "real physical phenomena", founded on physical laws based in the "hard sciences" and first principles. One sometimes hears claims that systems engineering lacks the equivalent phenomena-based theoretical foundations. In that telling, systems engineering is instead critically portrayed as emphasizing (1) process and procedure, (2) critical and systems thinking and good writing skills, and (3) organizing and accounting for information and risk in particular ways—valuable, but not as based on an underlying "hard science".

That view is perhaps understandable, given the initial trajectory of the first 50 years of systems engineering (Adcock 2015, Checkland 1981, Walden et al. 2015, Wymore 1977). "Science" or "phenomenon" of generalized systems have for the most part been described on an intuitive or qualitative basis, with limited reference to a "physical phenomenon" that might be called the basis of systems science and systems engineering. Some systemic phenomena (for example, requisite variety, emergence of structure, complexity, chaos theory, etc.) have received attention, but it is challenging to argue that these insights have had as great an impact (yet) on the human condition and engineering practice as the broader STEM illustrations cited above for the most recent three centuries of physical sciences and mathematics. However, INCOSE's own stated vision (Beihoff et al. 2014) calls upon systems engineering for such a result.

Respectful of the contributions of those early thinkers in systems engineering, we also note that their contributions can in some cases be expressed as manifestations of the modeled system phenomenon described below, advancing the scientific foundations of systems engineering.

MBSE, PBSE: Enabling a Phase Change in Systems Engineering

In the case of systems engineering, a key part of the story is that the role that quantitative system models have played, or not played, during its initial history. Most recently, the broader INCOSE-encouraged role for model-based methods offers to eventually accelerate the "phase change" that the successful earlier history of science, mathematics, and other engineering disciplines suggest is now in progress.

Models are certainly not new to segments of engineering practice. However, we are representing an increasingly fraction of our overall understanding of systems, from stakeholder trade space, to required functionality and performance, to design, and to risk, using explicit and increasingly integrated system models. As in Newton's Day, this also puts pressure on the approaches to model representations, in order that they effectively represent, conveying enough, and not too much, about the key ideas concerning the real things they are intended to describe.

The progress of physical sciences did not arise from models that only could describe single unique instances of systems, but instead represented what came to be understood as more general patterns that recur across broad families of systems. Likewise, there is an increasing effort in systems engineering to recognize that these models must often describe patterns of similarity and variation. This recognition of recurring patterns is necessary both from the perspectives of science and economics. The increasing use of explicit model-based patterns in these representations is a part of this phase change (INCOSE Patterns WG 2015, INCOSE MBSE Initiative 2015). Pattern-based systems engineering (PBSE) as an extension of model-based systems engineering (MBSE) increases emphasis on representation.

This is a more significant change than just the emergence of standards for systems modeling languages and information technology (IT) toolsets, even though those are valuable steps. We need underlying model structures that are strong enough — remember physics before the calculus of Newton and Leibniz. As a test of "strong enough," we suggest the ability to have the kinds of impact on humankind summarized in Section 2 — beginning with clearer focus on what phenomena were being represented.

Although this challenge sounds sobering, we will next argue that it is not necessary for emerging systems models to "start from scratch" in their search for new system phenomena, and further argue that what is already known from the earlier phase change of Section 2 helps suggest what aspects of our systems models need to be strengthened during the phase change in systems engineering. PBSE further reminds us of a practical lesson from the STEM revolution. Once validated patterns emerge, we (mostly) need to learn and apply those patterns (laws, principles), not how to re-derive them from earlier knowledge. Examples include the *periodic table* and the *gas laws*. While it may be controversial, "learn the model, not modeling" is advice worth considering, in a time when modeling from scratch seems carry more excitement.

THE SYSTEM PHENOMENON

The perspective used in this paper defines a system as a collecting of interacting components, where interactions involve the exchange of energy, force, mass, or information, through which one component impacts the state of another component, and in which the state of a component impacts its behaviour in future interactions (Schindel 2011).

In this framework, all behaviour is expressed through physical interactions (Figure 6). This perspective emphasizes physical interactions as the context in which all the laws of the hard sciences are expressed (Schindel 2013a).

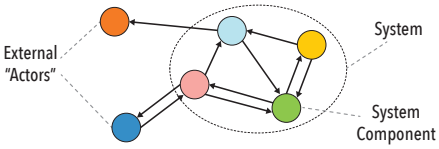


Figure 6. The system perspective

The traditional "phenomena" of the hard sciences are all cases of the following system phenomenon:

1. Each component has a specific behaviour during a given interaction type, determined by the component's state. (See (4) below for the source of that component's behavioural characteristics.)
2. The combined behaviours of the set of interacting components determine a combined system state space trajectory.
3. That trajectory is a collective property of the system components and interaction, and accordingly is not simply the description of possible behaviors of the individual components. For the systems discussed in this paper, by Hamilton's Principle (Levi 2014, Sussman et al. 2001, Hankins 2004), the emergent interaction-based behavior of the larger system is a "stationary" trajectory $X = X(t)$ of the action integral, based on the Lagrangian L of the combined system:

$$S[X] = \int_A^B L(X, \dot{X}, t)dt$$

4. The behavioural characteristics of each interacting component in (1) above are in turn determined by its internal ("subsystem") components, themselves interacting.

Reduced to simplest forms, the resulting equations of motion (or if not known or solvable, empirically observed paths) provide "physical laws" (or recurring observable behaviors) subject to scientific verification.

Instead of systems engineering lacking the kind of theoretical foundation that the "hard sciences" bring to other engineering disciplines, we therefore assert that:

- It turns out that all those other engineering disciplines' foundations are themselves dependent upon the system phenomenon and emerge from it.
- The related underlying math and science of systems (dating to at least Hamilton) provides the theoretical basis already used by all the hard sciences and their respective engineering disciplines.
- It is not systems engineering that lacks its own foundation—instead, it has been providing the foundation for the other disciplines! (Refer to Figure 1.)

Historical Domain Example 1: Chemistry

Chemists, and chemical engineers, justifiably consider their disciplines to be based on the "hard phenomena" of chemistry (Pauling 1960, Servos 1996):

- This perspective emerged from the scientific discovery and verification of phenomena and laws of chemistry.
- Prominent among these was the discovery of the individual chemical elements and their chemical properties, organized by the discovered patterns of the *periodic table*.
- Emerging understanding of related phenomena and behaviours included those of chemical bonds, chemical reactions, reaction rates, chemical energy, and conservation of mass and energy.
- Upon that structure grew further understanding of chemical compounds and their properties.

Even though these chemical phenomena and laws seemed very fundamental:

- All those chemical properties and behaviors are emergent consequences of interactions that occur between atoms' orbiting electrons (or their quantum

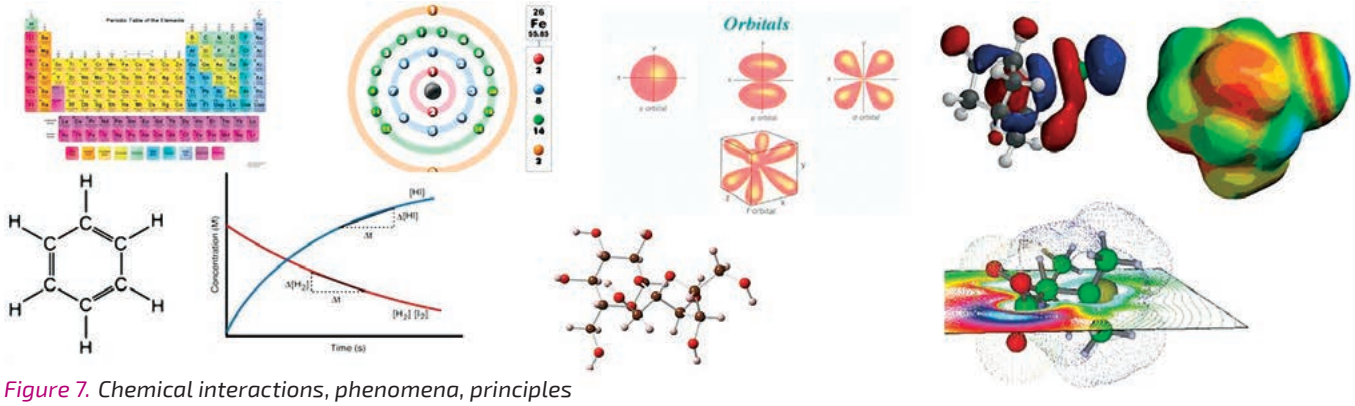


Figure 7. Chemical interactions, phenomena, principles

equivalents), along with limited properties (for example, atomic weights) of the rest of the atoms they orbit.
- These lower-level interactions give rise to the visible higher level chemical behaviour patterns that have their own higher level properties and relationships, expressed as "hard science" laws of chemistry.

So, we see that this illustrates:
- The "fundamental phenomena" of chemistry, along with the scientifically discovered / verified "fundamental laws / first principles" are in fact . . .
- Higher level emergent system patterns and . . .
- chemistry and chemical engineering study and apply those system patterns.

Historical Domain Example 2: The Gas Laws and Fluid Flow

Illustrated by Figure 8, the discovered and verified laws of gases and of compressible and incompressible fluid flow by Boyle, Avogadro, Charles, Gay-Lussac, Bernoulli, and others are rightly viewed as fundamental to science and engineering disciplines (Cardwell 1971).

However, all those fluid and gaseous properties and behaviors are emergent consequences of interactions that occur between atoms or molecules, the containers they occupy, and their external thermal environment. These lower-level interactions give rise to patterns that have their own higher-level properties and relationships, expressed as "hard sciences" laws. So, the "fundamental phenomena" of gases, along with the scientifically discovered and verified "fundamental laws and first principles" are in fact higher level emergent system patterns. And so, mechanical engineers, thermodynamicists, and aerospace engineers can study and apply these system patterns.

Examples from More Recent History

The practical point of this paper is to emphasize the constant emergence of new scientific and engineering disciplines, in domains arising from higher level system interactions. These include domains that have been important to society, even though they arose later than the more fundamental domains from which they spring. The discovery and exploitation of these higher-level phenomena, principles, and laws is important to future progress and innovation, including enterprises, careers of individuals, and society.

These more recent emergent domains, in which formal system patterns are being recognized as describing higher-level phenomena and laws, are illustrated by examples of Figure 9:

1. Ground vehicles: As in the dynamical laws of vehicle stability that enable vehicular stability controls (Guiggiani 2014).
2. Aircraft: Including the dynamical laws at the aircraft level that enable advanced aircraft design for dynamic performance and top-level flight controls (Pratt 2000).
3. Marine vessels: Facilitating the design of more efficient hulls and special purpose craft, as well as bulk transports (Perez et al 2007).
4. Biological regulatory networks: Advancing our understanding of immune reactions and other regulatory paths in connection with pathologies as well as therapies (Gene Regulation Wikipedia).

For example, in the case of ground vehicles, dynamical laws of vehicle stability arise from the interactions, modulated through control algorithms, of the distributed mass of the vehicle in motion with the driving surface, transmitted

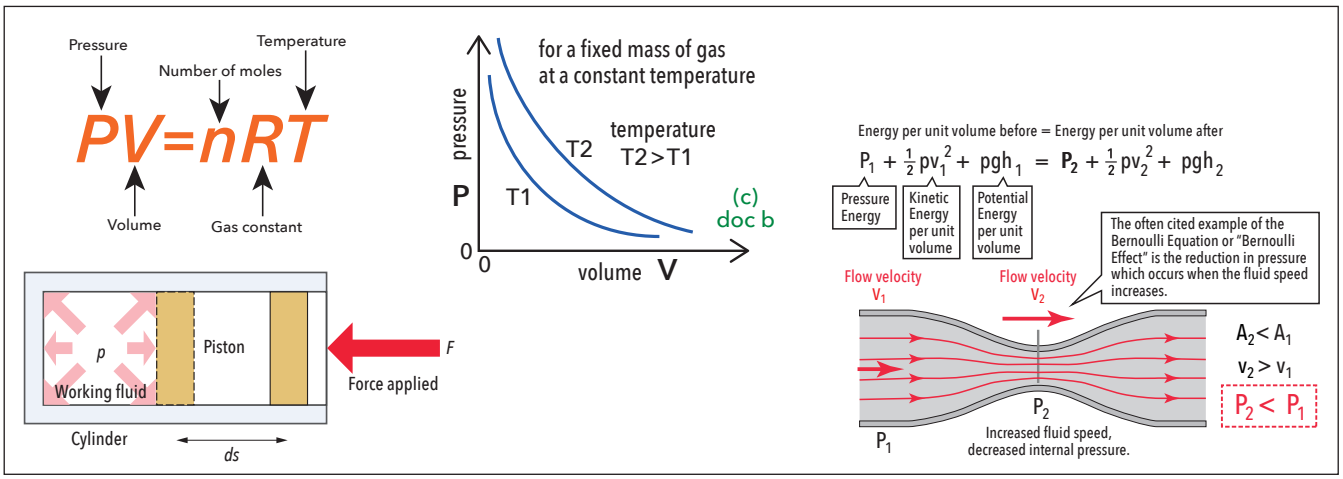


Figure 8. Gas, fluid interactions, phenomena, principles

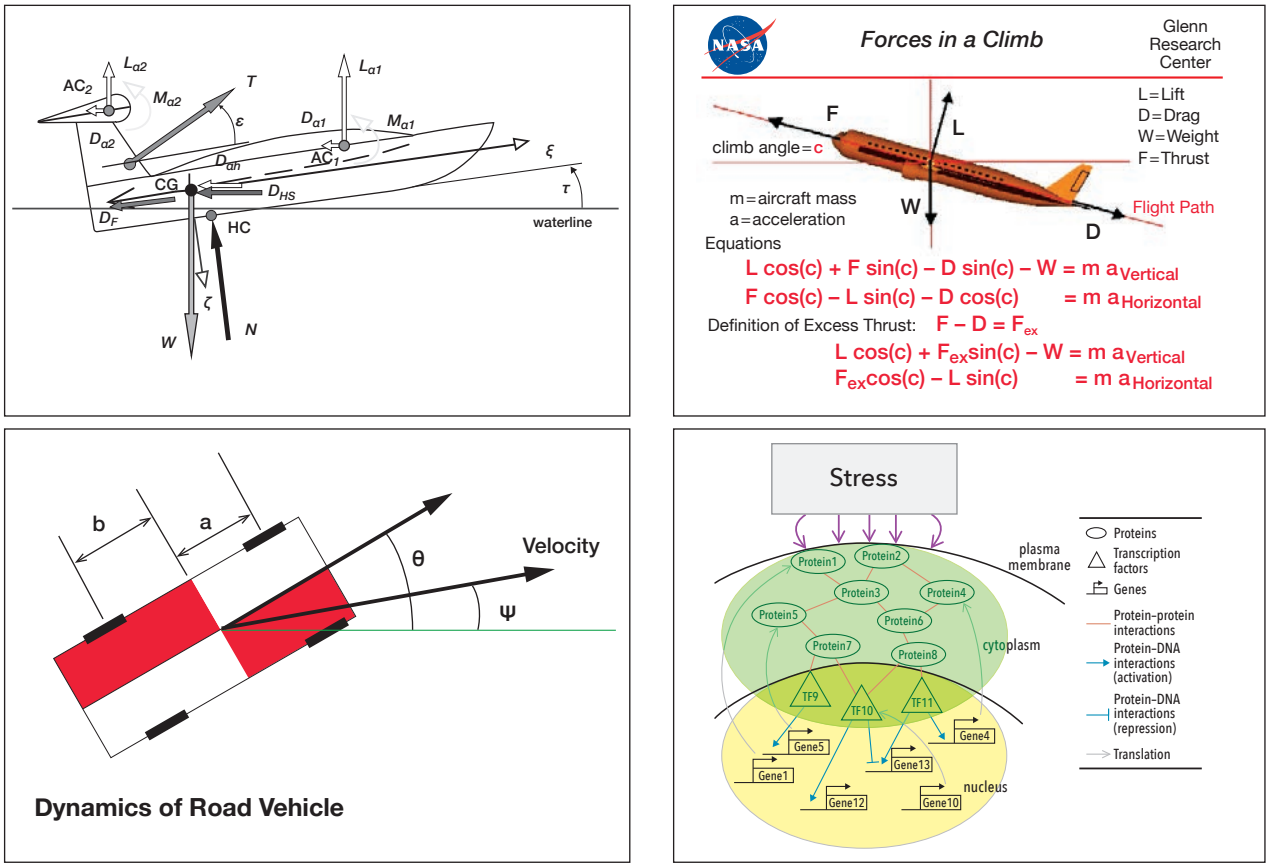


Figure 9. Ground and marine vehicles, aircraft, regulation in organisms

through tractional forces of braking, acceleration, or steering, as further impacted by road surface and tire conditions, along with other factors. It is the overall system interaction of all these domain elements that leads to emergent vehicular laws of motion.

Students of complexity (Cowan et al. 1994) will note that nonlinearity, the onset of chaos, and extreme interdependencies are not reasons to avoid representing the interactions manifesting that behaviour. Indeed, they provide further reasons to understand those very interactions.

Future Applications

Examples (Figure 10) that call out for improved future efficacy in systems engineering include:

1. Utility and other distribution networks: Society has come to depend on rapidly evolving, often global, networks for distribution of goods and services, in the form of materials, energy, communication, and information services. What are the network-level phenomena, laws, and principles of these networks, bearing on their effectiveness and resiliency (Perez-Arriaaga et al. 2013)?
2. Market systems, economies, and human-imposed regulatory frameworks: These systems clearly have direct impact on society and individuals. The "designed" systems of top-down regulation imposed upon them include such prominent examples as regulation of banking, securities markets, development of medical devices and compounds, and delivery of health care. What are the system-level phenomena, laws, and principles of these systems, bearing on their effectiveness and resiliency (Friedman 1980)?
3. Living ecologies: The emergent habitats of living things include rain forests, coral reefs, the human microbiome, and the biosphere as a whole. These demonstrate characteristics that include regulatory stability within limits, along with pathologies. What are the system-level phenomena, laws, and principles of these systems (MacArthur and Wilson 2001).
4. Health care delivery: These systems, including a number of important challenges, are much in the public eye. The very definition of effective health care is necessarily dynamic because of the evolving frontiers of medical science. The means of effectively delivering care, financing its costs, and

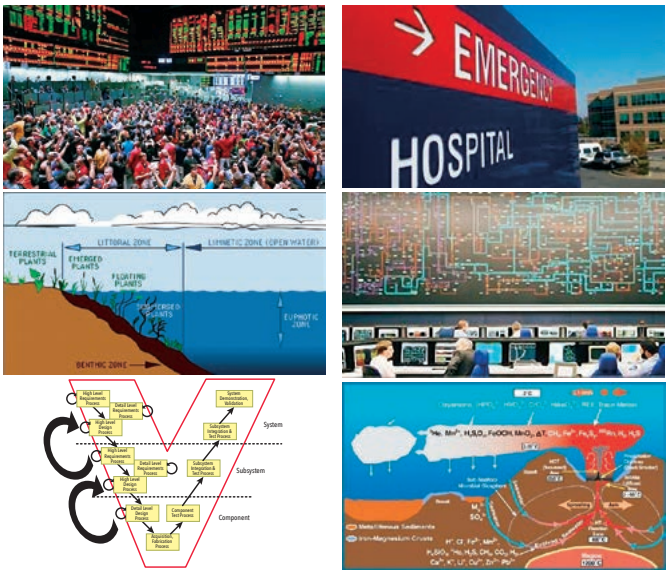


Figure 10. Domain systems of future interest

(Hippocratically) protecting patients from harm are all subject of study as to system-level phenomena and principles (Holdren et al. 2014).

5. Product development, general innovation, and related agility: This system domain is the "home court" of INCOSE and our systems engineering profession. While there is a large body of descriptions of the related systems, the study of these systems as modelled technical systems is mostly new or in the future. One such project is the INCOSE agile systems engineering life cycle model project (Braha et al. 2007, Schindel 2015, Schindel and Dove 2016, Hoffman 2015).

STRENGTHENING THE FOUNDATIONS OF MBSE

Like mechanics before Newton, the models of MBSE require a strengthened underlying framework to effectively describe the system phenomenon in the domains of practice. MBSE requires a strong enough underlying metamodel to support a phenomenon-based systems science.

As discussed in Schindel (2013a), interactions play a central role in that framework, inspired by Hamilton and three hundred years of pioneers in the emergence of science and engineering. Interactions are acknowledged by and can be modelled in some current system modelling frameworks, but typical practice and underlying structures need related improvement. Figure 11 illustrates a related, interaction-centric, extract from the S*Metamodel (Schindel 2011).

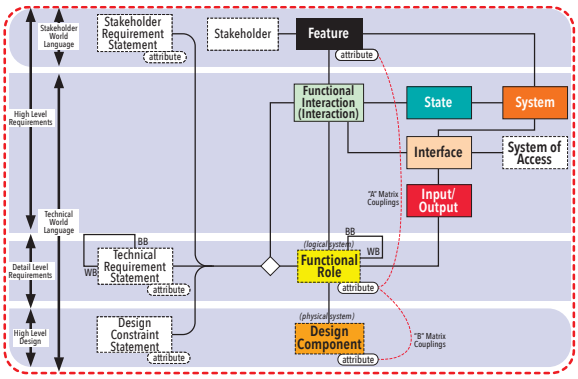


*Figure 11. Summary view of S*Metamodel*

This is something more than model semantics or ontology alone. It also means recognizing that the models we pursue are models of the real physical systems they are about, and not just models of information about business processes concerned with those systems. While that might seem obvious to the physical scientist, a different perspective than that is embedded in forty years of enterprise information system practice. In that history, the traditional (and relatively successful) paradigm is construction of information models that describe information transactions or documents (for example, purchase of air travel tickets). Symptomatic of that paradigm, today we still encounter MBSE models and human interpretations of them that include notions of databases, "calls," "methods," and other successful software notions that are not the same as modelling physical systems. Executable models add to this challenge. In the midst of this phase change, we live in interesting times.

An Illustration of Related Systems Engineering Impact: Design Review

As an example of the beneficial impact of this interaction-centric view of systems engineering, consider design review, where the system phenomenon appears front and center. Figure 12 is an extract from a guide to such a review in an MBSE setting (Schindel 2007). This diagram summarizes six questions relevant to reviewing whether a proposed system design will satisfy a set of technical requirements. Note Question 2, which compares the behaviour that emerges from interaction of its "white box" subsystems to the desired behaviour expressed by the system's "black box requirements."

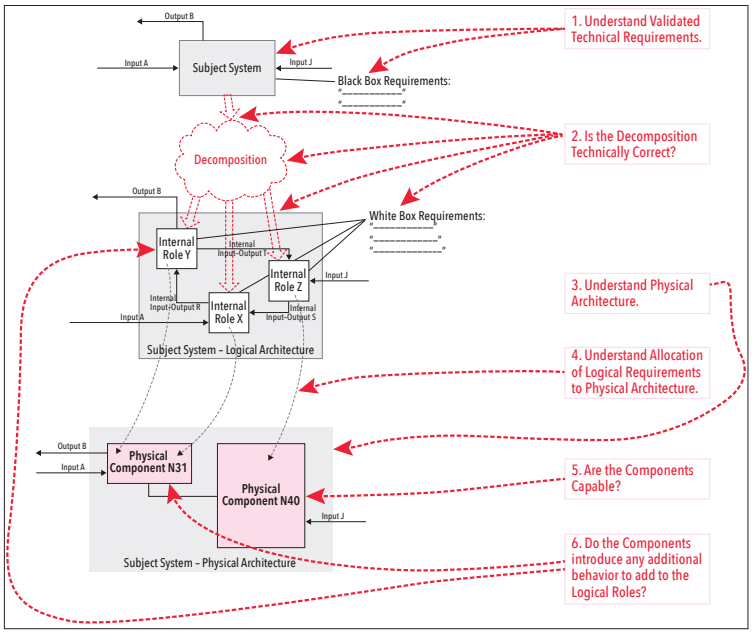


Figure 12. Related extract from MBSE design review guide

Whether viewed as composition (bottom up, emergence) or decomposition (top down), the ability to effectively answer Question 2 above is central to the design or design review process. Question 2 is about Hamilton's Principle in a specific domain setting. A verified library of knowledge of the related emergence or decomposition patterns that apply in an enterprises' or industry's or society's domains can be valuable. The capture and verification of such a library can be seen to be a form of system science in the tradition of the domain-specific hard sciences—whether the domain is lower level or high-level systems discussed above.

CONCLUSIONS AND IMPLICATIONS FOR FUTURE ACTION

1. Like the other engineering disciplines, systems engineering can be viewed as founded on "real" physical phenomena—the system phenomenon—for which experimentally verified, mathematically modeled hard science, laws, and first principles have existed for 150 years, dating to Hamilton, or earlier, to Newton.
2. Systems engineering not only has its own phenomenon, but the phenomena upon which the traditional engineering disciplines (civil, mechanical, chemical, electrical) are based can themselves all be seen to be derivable from the system phenomenon. (Note carefully that nothing about this suggests modeling behavior of an aircraft carrier from models of molecules—it simply notes that the same general interaction-based system phenomena is the basis of emergence of behavior at each higher system level.)
3. The system phenomenon supports the emergence of hard sciences, laws, and first principles for higher level systems of critical importance to the well-being of humankind.
4. Systems engineering, along with its related scientific foundations, is a young and still emerging discipline. The re-planting of systems en-

gineering in a model-based framework is an important step toward strengthening the discipline but requires a stronger model framework for that to occur, and the system phenomenon points the way to a key part of that framework.

Implications for future action include:

1. Beyond the scope of this paper, there are also other aspects of that strengthened modeling framework in need of attention. The purpose-oriented nature of engineering reminds us that a stronger representation of value, fitness space, and selection is a part of that framework (Schindel 2013b).
2. The INCOSE MBSE Patterns Working Group is practicing the PBSE representation of S*Patterns, which are MBSE models of recurring whole-system characteristics important to systems engineering. Participation in this INCOSE working group is invited (INCOSE Patterns WG). ■

REFERENCES

- Adcock, Rick, ed. "Guide to the Systems Engineering Body of Knowledge (SEBoK)." http://sebokwiki.org/wiki/Guide_to_the_Systems_Engineering_Body_of_Knowledge_(SEBoK)
- Arthur, W. Brian. 2009. *The Nature of Technology: What It Is and How It Evolves*. Free Press.
- Ashby, William Ross. 1956. *An Introduction to Cybernetics*. Wiley.
- Beihoff et al. 2014. "A World in Motion: Systems Engineering Vision 2025, International Council on Systems Engineering. San Diego, US-CA.
- Bertalanffy, L. von. 1969. *General System Theory: Foundations, Development, Applications*, George Braziller Inc.; Revised edition.
- Braha, D., A. Minai, Yaneer Bar-Yam, eds. 2006. *Complex Engineered Systems: Science Meets Technology*. New England Complex Systems Institute (NECSI), Cambridge, US-MA: Springer.
- Braha, Dan, and Yaneer Bar-Yam. 2007. "The Statistical Mechanics of Complex Product Development: Empirical and Analytical Results." *Management Science*, 53 (7): 1127–1145.
- Cardwell, D. S. L. 1971. *From Watt to Clausius: The Rise of Thermodynamics in the Early Industrial Age*. London, GB: Heinemann..
- Checkland, P. 1999. *System Thinking, System Practice*. Chichester, GB: Wiley.
- Cowan, George, David Pines, and David Meltzer. 1994. *Complexity: Metaphors, Models, and Reality*. Proceedings Volume XIX, Santa Fe Institute Studies in Science of Complexity, Santa Fe, US-NM: Addison-Wesley.
- Forbes, Nancy, and Basil Mahon. 2014. *Faraday, Maxwell, and the Electromagnetic Field: How Two Men Revolutionized Physics*. Amherst, US-NY: Prometheus Books.
- Friedman, Milton and Rose Friedman. 1980. *Free to Choose: A Personal Statement*. New York, US-NY: Harcourt.
- Guiggiani, Massimo. 2014. *The Science of Vehicle Dynamics: Handling, Braking, and Ride of Road and Race Car*. Dordrecht, NL: Springer.
- Hankins, T. 2011. *Sir William Rowan Hamilton*. Baltimore, US-MD: Johns Hopkins University Press.
- Holdren, John P., Eric S. Lander, et al. 2014. "Report to the President—Better Health Care and Lower Costs: Accelerating Improvement Through Systems Engineering." Executive Office of the President, President's Council of Advisors on Science and Technology, May. http://www.whitehouse.gov/ostp/pcast
- Hoffman, C. and W. Schindel. 2015. "Systems Engineering Community of Practice Social Network Pattern." Presented at the INCOSE 9th Annual Great Lakes Regional Conference, Cleveland, US-OH, 23-25 October.
- Holland, John H. 1998. *Emergence: From Chaos to Order*. New York, US-NY: Perseus, 1998.
- INCOSE MBSE Initiative Patterns Working Group. http://www.omgwiki.org/MBSE/doku.php?id=mbse:patterns:patterns.
- INCOSE Patterns Working Group. n.d. "MBSE Methodology Summary: Pattern-Based Systems Engineering (PBSE), Based On S*MBSE Models" V1.5.5A. http://www.omgwiki.org/MBSE/doku.php?id=mbse:pbse.
- Kauffman, Stuart. 2000. *Investigations*, Oxford, GB: Oxford University Press.
- Levi, M. 2014. *Classical Mechanics with Calculus of Variations and Optimal Control*. Providence, US-RI: American Mathematical Society.
- MacArthur Robert H., and Edward O. Wilson. 2001. *The Theory of Island Biogeography*, Princeton, US-NJ: Princeton University Press.
- Mokyr, Joel. 2009. *The Enlightened Economy: An Economic History of Britain 1700-1850*, New Haven, US-CT: Yale University Press.
- Morris, Charles R. 2012. *The Dawn of Innovation: The First American Industrial Revolution*. New York, US-NY: Public Affairs.
- Pauling, L. 1960. *The Nature of the Chemical Bond and the Structure of Molecules and Crystals: An Introduction to Modern Structural Chemistry* 3rd edition. Ithaca, US-NY: Cornell University Press.
- Perez, Tristan, and Thor I. Fossen. 2007. "Modelling and Simulation of Marine Surface Vessel Dynamics." Tutorial, *IFAC Conference on Control Applications in Marine Systems*, Bol, HR.
- Pérez-Arriaga, Ignacio, et al. 2013. "From Distribution Networks to Smart Distribution Systems: Rethinking the Regulation of European Electricity DSOs." *THINK Project Final Report*, European University Institute, Florence, IT.
- Pratt, Roger W., ed. 2000. *Flight Control Systems: Practical Issues in Design and Implementation*. IEE Control Engineering, Padstow, GB: TJ International.
- Prigogine, Ilya. 1980. *From Being to Becoming: Time and Complexity in the Physical Sciences*. US: Freeman.
- Rogers, Everett M. 2003. *Diffusion of Innovations* Fifth Edition. New York, US-NY: Free Press.
- Schindel, W. 2015. "Introduction to the Agile Systems Pattern: An MBSE-Based System Pattern, with Implications for Agile Modeling." Presented at INCOSE IW2015 MBSE Workshop Breakout Session: Agile Modeling and Modeling Agile Systems, 24 January. http://www.omgwiki.org/MBSE/doku.php?id=mbse:incose_mbse_iw_2015:breakout_out_ses sion_agile_modeling.
- Schindel, W. 2013a. "System Interactions: Making The Heart of Systems More Visible," Presented at the INCOSE 7th Annual Great Lakes Regional Conference, West Lafayette, US-IN, 5-6 October.
- Schindel, W. 2013b. "Systems of Innovation II: The Emergence of Purpose." Paper presented at the 23rd Annual International Symposium of INCOSE, Philadelphia, US-PA, 24-27 June.
Schindel, W. 2011. "What Is the Smallest Model of a System?" Paper presented at the 21st Annual International Symposium of INCOSE, Denver, US-CO, 20-23 June.

> continued on page 32

Explicating System Value through First Principles: Re-Uniting Decision Analysis with Systems Engineering

Troy Peterson, tpeterson@systemxi.com; and **Bill Schindel,** schindel@ictt.com

[Editor: This paper for systems engineering foundations refers to the *Systems Engineering Handbook* 4th edition (Copyright 2015 by the International Council on Systems Engineering), ISO 15288:20915, and the Systems Engineering Vision 2020 published by INCOSE in 2007.]

■ **ABSTRACT**
System complexity continues to grow, creating many new challenges for engineers and decision makers. To maximize value delivery, "both" systems engineering and decision analysis are essential. The systems engineering profession has had a significant focus on improving systems engineering processes. While process plays an important role, the focus on process was often at the expense of foundational engineering axioms and their contribution to system value. As a consequence, systems engineers were viewed as process developers and managers versus technical leaders with a deep understanding of how system interactions are linked to stakeholder value. With the recent shift toward model-based systems engineering (MBSE), systems engineering is "getting back to basics," focusing on value delivery via first principles, using established laws of engineering and science. This paper describes how pattern-based systems engineering (PBSE), as outlined within INCOSE's model-based systems engineering (MBSE) initiative, explicates system value through modeling of first principles, re-uniting systems engineering and decision analysis capabilities.

INTRODUCTION

Complexity in systems and decisions: Today, the scope of engineering efforts often rapidly expands to include more and more external interactions. Additionally, within a defined system boundary, systems are becoming significantly more interconnected. Collectively this is accelerating the number of interactions engineers need to understand and manage. This increase and the associated challenges show no sign of abatement as shown in Figure 1 which depicts the explosion of the Internet of things (IoT). IoT is a significant contributor to the increase in connectedness and system complexity, and we are still only in the formative stages of this exponential growth. Furthermore, this interconnected phenomenon is ubiquitous, occurring across domains and with systems we use every day.

In addition to the increased density of interactions, the pace of contextual change is also increasing. The contextual dynamics have the effect of continually altering a system's fitness and value. This further complicates matters, adding the challenge to design into systems the necessary flexibility and agility, giving rise to a more stochastic view of design rather than a more traditional steady state, deterministic perspective.

This context obviously brings about many challenges for engineers and decision makers, which extend beyond the technical domain. Given the complexity and web of interactions, a decision that may appear simple at first could have significant strategic, social, political, and economic impact. Where an engineer or manager's intuition may have been sufficient decades ago – today, when trying to consider of second, third and fourth order impacts, the complexity can quickly overwhelm any one person or even a highly capable team.

In his book *Notes on the Synthesis of Form*, Christopher Alexander (1964) articulated this context eloquently over 50 years ago. The following statements are excerpts from his book:

> *Today more and more design problems are reaching insoluble levels of complexity*
>
> *At the same time that problems increase*

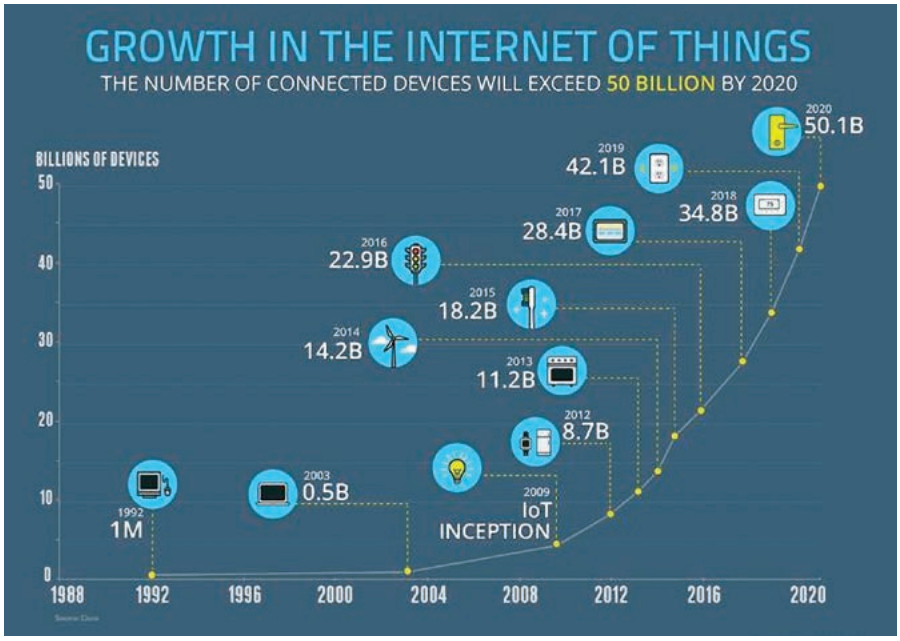


Figure 1. Explosive growth in the Internet of things (Cisco n.d.)

in quantity, complexity and difficulty, they also change faster than before

Trial-and-error design is an admirable method. But it is just real world trial and error which we are trying to replace by a symbolic method. Because trial and error is too expensive and too slow

These statements are more applicable today than they were 50 years ago and they will be even more applicable 50 years from now. Consequently, approaches which leverage symbolic method, speed feedback and iterations, build in agility and ensure a holistic view are essential. One important aspect of ensuring our methods emphasize such results is to better couple the decision making and innovation processes and related models.

History and a call for a new view: The history of systems engineering has strong ties to fundamental engineering disciplines, the sciences and to mathematical modeling and managerial decision support (management sciences)–often referred to as decision analysis, industrial engineering, or operations research. So, in many ways a discussion of how to integrate these disciplines is a return to an early foundational element of systems engineering.

To help address the complexity previously outlined and to better re-integrate systems engineering and decision analysis many efforts are underway within industry, the government and non-profits. For example, a working group within INCOSE focuses on decision analysis with the purpose of advancing the state of the practices, education, and theory of decision analysis and its relationship to other systems engineering disciplines. The council of engineering systems universities (CESUN) is another example which was formed to address the great challenges posed by large-scale, interconnected, and therefore highly complex and dynamic, socio-technical systems. The excerpt from the CESUN website which follows articulates the contributions of systems engineering and decision analysis to engineering systems.

As many engineers began to delve deeper and deeper into science, some others stressed the design perspective and explored how to solve the problems arising from greater technical complexity. Operations research, systems and decision analysis, industrial engineering, systems engineering—these all contributed to the expansion of engineering—but at a certain point there was a recognition that some of the greatest challenges were precisely where the technical systems had their interfaces with people, policies, regulations, culture, and behavior (CESUN n.d.).

This excerpt also calls out the expanded and new view at the "… interfaces with people, policies, regulations, culture and behavior." This perspective brings with it a diverse set of stakeholders and an expanded view of value. To achieve value delivery in this new view we must have an improved coupling of systems engineering and decision analysis. The disciplines are absolutely complementary with systems engineering providing an overall approach to systematically innovate and decision analysis providing a systematic approach to think about, experiment with, and analyze complex problems or opportunities throughout the innovation process.

To fully integrate these disciplines the third bullet from Alexander noted above makes an important observation about the use of "…symbolic method. Because trial and error is too expensive and slow." This brings us first to the use of models and model-based systems engineering (the symbolic part) and then to the agile systems engineering life cycle pattern (the sped-up "trial and error" part).

One might at first assume that this sets up a rivalry between symbolic model-based analysis and simulation versus waiting for post-development market judgment. However, the agile systems engineering life cycle pattern (Schindel and Dove 2016) reminds us of the limits of symbolic models and provides a "middle way:" Using "the market" throughout the development cycle, moving "who makes the decisions" of development-time decision analysis, to include the ultimate decision-maker—the stakeholder.

RE-UNITING DECISION ANALYSIS WITH SYSTEMS ENGINEERING

Many frameworks group, categorize or connect decision analysis with systems engineering. This is true within the overview of system engineering provided by the Defense Acquisition University shown in Figure 2, and with the INCOSE *Systems Engineering Handbook* (Walden et al. 2015) as shown in Figure 3. Figure 4 is from the agile systems engineering life cycle management (ASELCM) pattern (Schindel and Dove 2016).

The process view: The Defense Acquisition University states that the decision analysis process transforms a broadly stated decision opportunity into a traceable, defendable, and actionable plan. Furthermore, that it is performed at all systems levels and across the life cycle. The DAU outlines decision analysis integration specifically with the process areas of technical planning, assessment, stakeholder requirements, requirements analysis and architecture design all shown in Figure 2. INCOSE also notes the decision management process, which includes decision analysis, integrates with all other systems engineering processes in its system life-cycle process N^2 chart found in the Appendix A of the *Systems Engineering Handbook* (2015). Figure 3 provides a view of the system life cycle processes aligned with ISO 15288 and INCOSE's *Systems Engineering Handbook*. The ASELCM reference boundary diagram shown in Figure 4 contains the same systems engineering processes in an abstract form

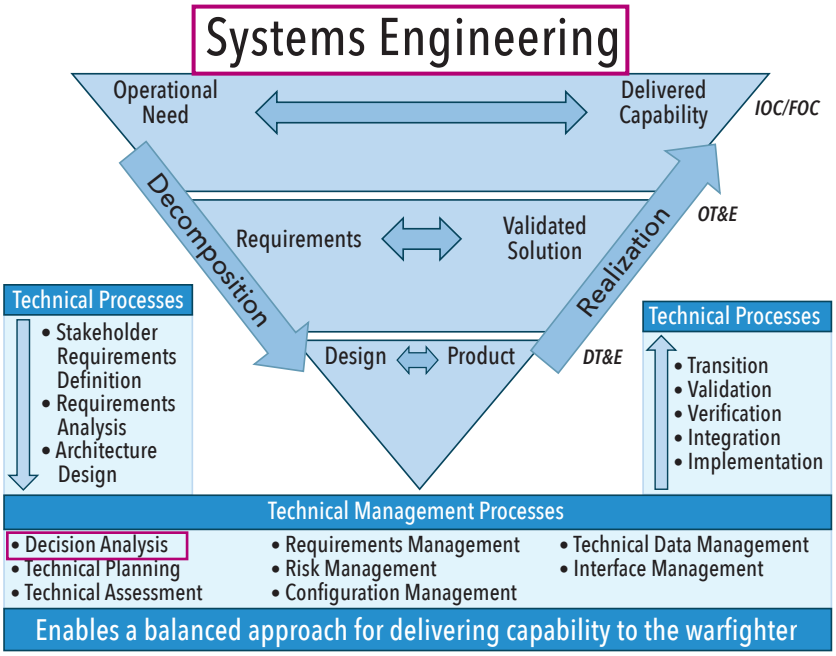


Figure 2. DAU systems engineering diagram (DAU n.d.)

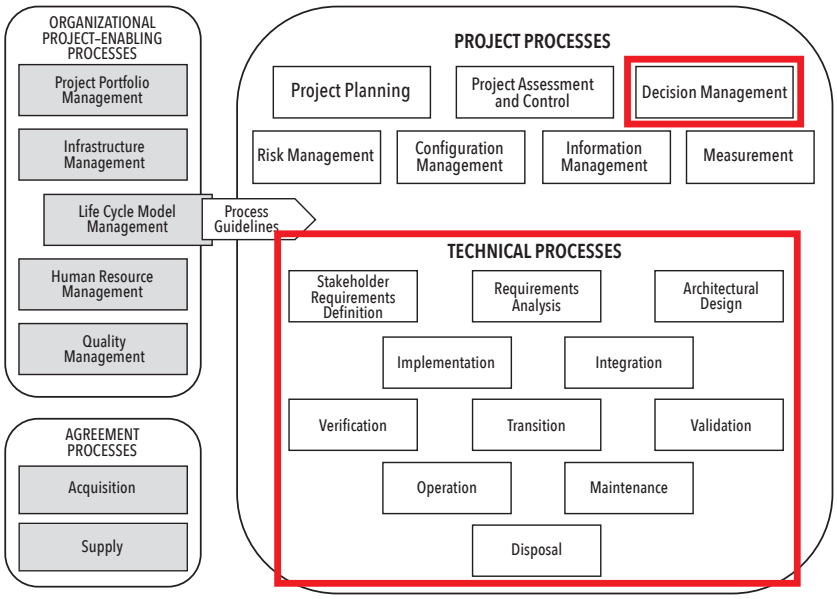


Figure 3. INCOSE system life-cycle processes overview per ISO 15288.

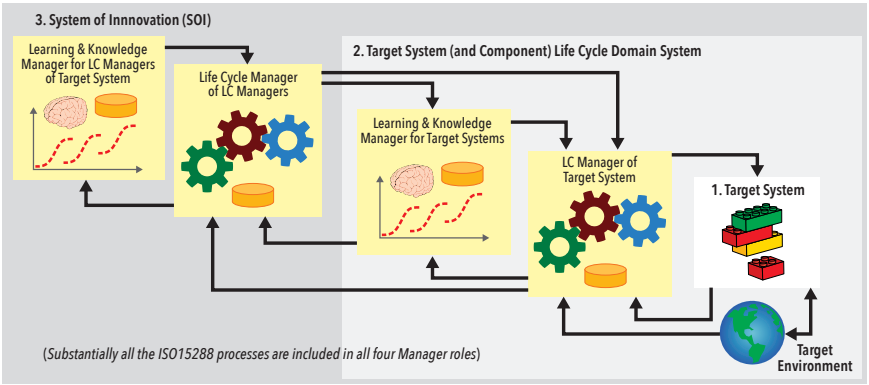


Figure 4. Iconic view of the agile systems engineering life cycle management (ASELCM) reference boundaries (Schindel and Dove 2016)

as life cycle system management processes (shown in yellow boxes) however it also introduces the target system, target system models and the target environment all of which are essential when considering how to fully integrate systems engineering and decision analysis.

While these views and their associated processes play an essential role in engineering complex systems this view alone is insufficient. Taken to the extreme, some focused solely on systems engineering processes omitting an essential aspect of fully integrating systems engineering and decision analysis found in how system interactions deliver value. As a consequence, systems engineers at times have been viewed solely as process developers and managers versus technical leaders with a deep understanding of how system interactions are connected to stakeholder value.

For well over a decade the systems engineering profession has had a significant focus on improving systems engineering processes – as illustrated by CMMI (2010) and ISO 15288. Connections between the systems engineering and decision analysis exist at a high level as shown in Figures 2 and 3 as well as within many more detailed process architectures. These connections are important and help program teams manage the complex system of innovation. However, there is a deeper need in connecting these disciplines, both more deeply and in a more explicit way to ensure value delivery.

Models of process vs models of systems: Process integration is important and helpful, but alone it is not sufficient to manage the complexity in systems today—in fact it can become nearly impossible to avoid unintended consequences without detailed models of the target system. Much of the integration effort of systems engineering and decision analysis has been focused on process — the infrastructure for information about the system of interest. It has not been, however, as focused on the information that passes through the process about the target system.

With the recent shift toward model-based systems engineering (MBSE), the systems engineering discipline is "getting back to basics" and back to the foundational engineering axioms built upon first principles and established laws of science and engineering. This focus is more aligned with the genesis of classical mechanics, beginning with Newtonian interactions and their emergent properties, so that the whole is greater than the sum of the parts. First principles as used here mean interactions of force, energy, mass flow, and information flow. This includes established laws of physics and emerging higher-level laws. For

example, this could include well-formed and understood interactions and patterns within domains such as automotive, healthcare, energy, and others (Schindel 2016).

Using models to connect first principles to stakeholder value is first accomplished through an explicit connection in the metamodel of how we model systems. More specifically by expressing and directly connecting both stakeholder value and system interactions. At risk in this connection is misunderstanding the value to first principle connections, for example having a narrowly defined stakeholder space and omitting one or more stakeholders. This risk is addressed in the agile systems engineering life cycle pattern, which expresses stakeholder value as demonstrated by selection interactions in the target system environment which more frequently incorporate feedback and design iterations.

MODEL-BASED SYSTEMS ENGINEERING

INCOSE defines model-based systems engineering (MBSE) as "the formalized application of modeling to support system requirements, design, analysis, verification and validation activities beginning in the conceptual design phase and continuing throughout development and later life cycle phases…." (INCOSE 2007). The Object Management Group's MBSE wiki notes that "Modeling has always been an important part of systems engineering to support functional, performance, and other types of engineering analysis" (OMG n.d.).

The application of MBSE has increased dramatically in recent years and is becoming a standard practice to help manage the complexity seen in systems today. MBSE has been enabled by the continued maturity of modeling languages such as SysML® and significant advancements made by tools vendors. These advancements are improving communications and providing a foundation to integrate diverse models. MBSE is often discussed as being composed of three fundamental elements – tool, language, and method. The third element, method, however, has not always been given proper consideration. Because the language and tool are relatively method independent, it is methodology which further differentiates the effectiveness of any MBSE approach and its ability to help manage the complex and interrelated functionality of today's highly interconnected systems. For the approach discussed in this paper, the "methodology" includes not only process as discussed in the previous section in accordance with ISO 15288, INCOSE, DAU, or others, but more significantly the very concept of the underlying system information those processes produce and consume, independent of modeling language and tools.

PATTERN-BASED SYSTEMS ENGINEERING

Pattern-based systems engineering (PBSE) as outlined within INCOSE's model-based systems engineering (MBSE) initiative (OMG n.d.), is a methodology which formalizes historical pattern efforts using explicit, re-usable, configurable S*Models (S*Patterns). Moreover, it explicates system value through an understanding of system interactions and their projection onto value space (features). Pattern-based systems engineering (PBSE) can address 10:1 more complex systems with 10:1 reduction in modeling effort, using people from a 10:1 larger community than the "systems expert" group, producing more consistent and complete models sooner. These dramatic gains are possible because projects using PBSE get a "learning curve jumpstart" from an existing pattern and its previous users, rapidly gaining the advantages of its content, and improving the pattern with what is learned, for future users. The major aspects of PBSE have been defined and practiced for many years across a number of enterprises and domains. To increase awareness of the PBSE approach, two years ago INCOSE started a patterns challenge team (now the Patterns Working Group) within the INCOSE MBSE initiative.

The term "pattern" appears repeatedly in the history of design, such as civil architecture, software design, and systems engineering (Alexander 1977, Gamma et al. 1995, and Cloutier 2008). These are all similar in the abstract, in that they refer to regularities that repeat, modulo some variable aspects, across different instances in space or time. However, the PBSE methodology referred to by this paper is distinguished from those cases by certain important differences:

1. S*Patterns are model-based: We are referring here to patterns represented by formal system models, and specifically those which are re-usable, configurable models based on the underlying S*Metamodel. (By contrast, not all the historical "patterns" noted above are described by MBSE models.)
2. Scope of S*Patterns: We are referring here to patterns which will usually cover entire systems, not just smaller-scale element design patterns within them. For this reason, the typical scope of an S*Pattern applications may be thought of as re-usable, configurable models of whole domains or platform systems—whether formal platform management is already recognized or not. (By contrast, most of the historical "patterns" noted above describe smaller, reusable subsystem or component patterns.) S*Patterns are similar to architectural frameworks, although they contain more information.

Fundamental to pattern-based systems engineering is the use of the S*Metamodel (summarized by Figure 5), a relational/object information model used in the Systematica™ methodology to describe requirements, designs, and other information in S*Models such as verification, failure analysis, etc. (Schindel and Peterson 2013). A metamodel is a model of other models—a framework or plan governing the models that it describes. These may be represented in SysML®, database tables, or other languages. As an MBSE enabled approach PBSE can be implemented across multiple third-party commercial-off-the-

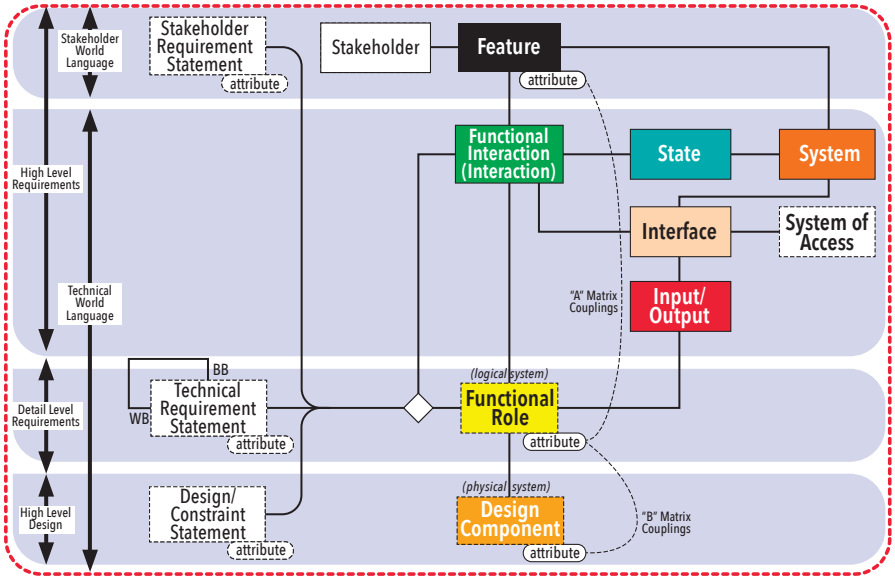


*Figure 5. A summary view of the S*Metamodel*

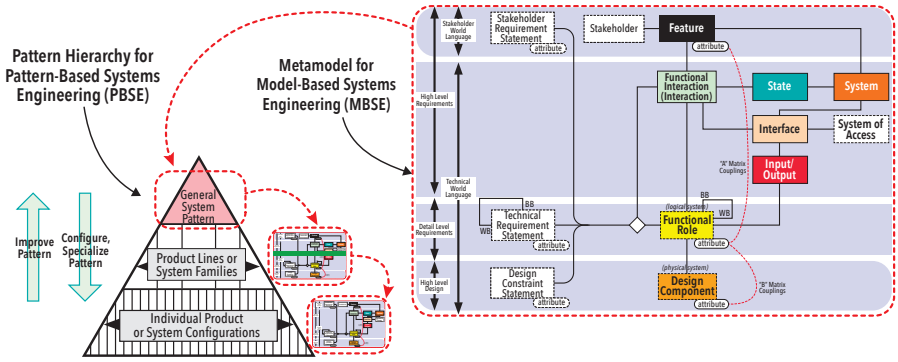


Figure 6. Pattern hierarchy for PBSE

shelf (COTS) tools and languages (that is, product line management PLM systems, modeling tools, architecture tools, databases, SysML, IDEF).

Specifically, an S*Pattern is a re-usable, configurable S*Model of a family of systems (product line, set, ensemble etc.) as shown in Figure 6.

Over several decades, pattern-based systems engineering has been developed and practiced across a range of domains, including carrier grade telecommunications, engines and power systems, automotive and off-road heavy equipment, military and aerospace, medical devices, pharmaceutical manufacturing, consumer products, and advanced manufacturing systems.

Engineers in these and many other domains spend resources developing or supporting systems that virtually always include major content from repeating system paradigms at the heart of their business (for example, core ideas about airplanes, engines, switching systems, etc.). Despite this, the main paradigm apparent in most enterprises to leverage "what we know" is to build and maintain a staff of experienced technologists, designers, application engineers, managers, or other human repositories of knowledge.

The physical sciences are based upon the discovery of regularities (patterns), which we say express laws of both nature and systems value markets. Although re-usable content has some history in systems engineering, there is less recognition of a set of "Maxwell's Equations" or "Newton's Laws" expressing the nature of the physical world, as the basis of those systems patterns. If electrical engineering and mechanical engineering disciplines have physical law at their foundation, why cannot systems engineering do the same?

By contrast, the S*Metamodel is focused on the very physical interactions that are the basis of the physical sciences, and which we assert are at the heart of the definition of system (in this methodology) as a collection of interacting components. The S*Patterns that arise from the explicit representation of physical Interactions re-form the foundation of system representations to align more explicitly with the physical sciences.

At its very foundation, the ASELCM pattern of PBSE links decision analysis and systems engineering ensuring system configurations are directly traceable and driven by stakeholder values. PBSE explicates system value via a formal model of interactions, whether force, mass flow, energy or information exchanges which are foundational to science and to the first principles of system design and market responses.

SYSTEM VALUE – STAKEHOLDER FEATURES

System value is measured by the selection interactions of stakeholders or their representatives; in the S*Metamodel these values are expressed explicitly as features. In the ASELCM pattern, these selections are as explicit as the (other) interactions of the system of interest. Features and their associated attributes contain the value space for a system of interest codified as formalized stakeholder needs/values. The connection between stakeholders and features is clear within the S*Metamodel shown in Figure 5. Features are shown at the top of the figure using a black box. Figure 7 reformats and displays just a portion of the S*Metamodel clearly annotating the classes from which we derive system value. Features are parameterized by feature attributes which provide a measure of value – including all stakeholder measures of effectiveness (MOEs). Within Figures 5 and 7 these feature attributes are represented by a white elongated oval adjacent to the black feature box, or by other symbology in SysML or other language.

As outlined in the introduction, just as the system boundary has broadened, the set of stakeholders and their respective values

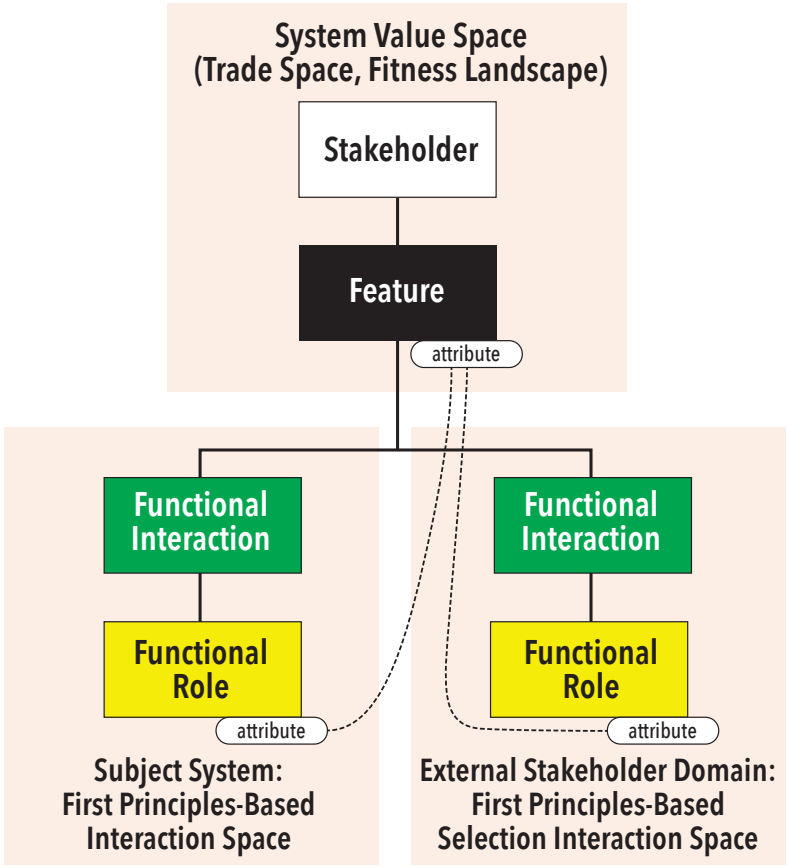


*Figure 7. S*Agile systems engineering life cycle pattern extract, highlighting system value which is generated via interactions – the first principles of engineering and science*

must also be broadened. It is important to note that stakeholders include all classes of stakeholders and not just those who may purchase or use a product or system of interest. Stakeholders include shareholders, manufacturers, society, and others. Every trade off or decision which sets the direction of a system design is a value judgment (selection interaction) from the perspective of one or more stakeholders. Given this view it is absolutely necessary to have a holistic view and identify the full complement of stakeholders.

In fact, it is the omission of stakeholders early in a systems program that often leads to costly rework, redesign, failures in system validation and sometimes program cancelation. When feature space is mature and expansive it can significantly reduce technical and programmatic risk. While ensuring the set of stakeholders is comprehensive it should not be assumed however that all stakeholders and their associated values are equal.

Since feature space contains the full complement of stakeholder values (the fitness landscape) it contains the entire trade space for the design and development of systems. This includes the full breadth and hierarchical depth of value including objectives and measures, weights, and rationale prescribed in texts focused on decision analysis (Keeney 1992, Clemen and Reilly 2001, and the Defense Acquisition Guidebook References). With stakeholders and their features well understood the features are used to configure systems that conform to the selections and the dialing in of their associated attributes.

Feature space is where selection-based decision analysis occurs. It is used as the basis of analysis and defense of all decision-making including optimization and trade-offs. This gives rise to the next class of information in Figures 5 and 7 which delivers system value – functional interactions.

FIRST PRINCIPLES – FUNCTIONAL INTERACTIONS

Functional interactions are what define a system (a group of interacting, elements forming a complex whole) and through which the system delivers value. Functional interactions involve the exchange of forces, mass, energy, or information. When we think of these fundamental exchanges, it brings to mind the work one would become intimately familiar in the study of physics, chemistry, mechanics, and many other engineering, science, or mathematics. These exchanges return us to the first principles of these disciplines and how they apply to the systems we design and develop. Additionally, as our understanding grows within a particular domain or with a specific type of system, we often begin to learn the first principles of these systems which are also expressed as interactions. These interactions can be at the component, subsystem or system level, and especially with the external environment of a system of interest.

Alternatives as outlined within decision analysis are the options to evaluate against decision criteria or the objectives and measures. Functional roles, displayed as the yellow box in Figures 5 and 7, are solution neutral logical roles which participate in an interaction. An identified functional interaction may be implemented by various combinations of functional roles. This gives rise to many alternatives when making traditional functional allocations.

In Figure 7 the feature and functional role attributes are coupled as shown by the dotted line. This particular coupling qualifies the fitness or trade space. The feature attribute defines the measure of effectiveness and the functional role attribute provides a means and measure of value delivery (level of performance) depending upon the selection of design components filling the functional role.

Parameterization and configuration: S*Models are intended to establish modeled feature sets for all stakeholders. This (features) portion of an S*Pattern is then used to configure the pattern for individual applications, product configurations, or other instances. It turns out that the variation of configuration across a product line is always for reasons of one stakeholder value or another, so feature selection becomes a proxy for configuring the rest of an S*Pattern into a specifically configured instance model.

Because S*Features and their feature attributes (parameters) characterize the value space of system stakeholders, the resulting S*Feature configuration space becomes the formal expression of the trade space for the system. It is therefore used as the basis of analysis and defense of all decision-making, including optimizations and trade-offs. The S*Feature space also becomes the basis of top-level dashboard model views that can be used to track the technical status of a project or product. All "gaps" and "overshoots" in detailed technical requirements or technologies are projected into the S*Feature space to understand their relative impact.

As illustrated by the "down stroke" in Figure 6, a generic S*Pattern of a family of systems is specialized or "configured" to produce an S*Model of a more specific system, or at least a narrower family of systems. Since the S*Pattern is itself already built out of S*Metamodel components, for a mature pattern the process of producing a "configured model" is limited to two transformation operations:

1. Populate: Individual classes, relationships, and attributes found in the S*Pattern are populated (instantiated) in the configured S*Model. This can include instances of features, interactions, requirements, design components, or any other elements of the S*Pattern. These elements are selectively populated, as not all necessarily apply. In many cases, more than one instance of a given element may be populated (for example, four different seats in a vehicle, five different types of safety hazard, etc.). Population of the S*Model is driven by what is found in the S*Pattern, and what features are selected from the S*Pattern, based on stakeholder needs and configuration rules of the pattern, built into that pattern.
2. Adjust values of attributes: The values of populated attributes of features, functional roles/technical requirements, and physical components are established or adjusted.

This brings into sharp focus what are the fixed and variable aspects of S*Patterns (sometimes also referred to as "hard points" and "soft points" of platforms). The variable data is called "configuration data." It is typically small in comparison to the fixed S*Pattern data. Since users of a given S*Pattern become more familiar over time with its fixed ("hard points") content (for example, definitions, prose requirements, etc.), this larger part is typically consulted less and less by veterans, who tend to do most of their work in the configuration data (soft points). That data is usually dominated by tables of attribute values, containing the key variables of a configuration. Since this is smaller than the fixed part of the pattern, in effect the users of the pattern experience a "data compression" benefit that can be very significant, allowing them to concentrate on what is or may be changing (Schindel 2011a).

Just as feature attributes parameterize stakeholder values, functional role attributes parameterize technical behavior. The coupling of these attributes shown in Figures 5 and 7 provides a model-based approach to coupling the first principles of engineering and science with stakeholder value. It is through this coupling that pattern-based systems engineering explicates system value through first principles.

The agile system life cycle pattern: INCOSE is currently executing the 2015-2016 agile systems engineering life cycle model (ASELCM) project (Schindel and Dove 2016). Working across a series of North American and European enterprises

and industries, this discovery project is articulating and validating the ASELCM Pattern mentioned in this paper, in the form of a formal S*Pattern.

The ASELCM pattern explicates the points summarized in this paper, including:

1. The deeper re-integration of decision analysis and systems engineering, with the decisions shared between "internal" decision-makers and agile-measured "external" stakeholder representatives, whose selection behaviors are studied as a faster and surer path to good decisions.
2. The use of explicit MBSE models to express life cycle system requirements, design, generated from MBSE patterns by configuration and reconfiguration, as the environment changes in non-deterministic ways, and as a point of accumulation of learning.

CONCLUSIONS

System complexity and interconnectedness continue to rapidly increase, making systems development extremely challenging. Additionally, the context in which developed systems operate is continually changing, altering the fitness and value delivered systems provide. The systems engineering discipline has made many great improvements through process definition and integration. While these improvements have enabled and structured innovation, they are not sufficient to overcome the outlined challenges, which are likely to only increase over time. Our traditional development activities must be revisited and enhanced to manage significant complexity, nth order impacts, highly dynamic contexts, complicated decisions and significant ambiguity.

An important aspect to an improved approach is to better integrate decision analysis and systems engineering and to leverage "symbolic method" (to the extent that symbolic analysis and simulation are sufficient) while also improving ability to capture stakeholder and market judgments without undue delay (to the extent that empirical experiment is also required). This leads us to modeling methods and the promise provided by model-based systems engineering. As a particular MBSE methodology, PBSE is particularly well suited to model complex systems. With interactions and features at the core of the S*Metamodel, PBSE focuses the engineering effort on how systems fundamentally provide value. It couples system value, experienced by stakeholders as features, with the first principles of engineering and science, expressed as functional interactions, making for a strengthened systems engineering approach. This approach also shifts the focus from the innovation process to the information passing through the process, which describes the system of interest, which ultimately determines the level of value provided to stakeholders. The explicit coupling within the modeling approach permits rapid iteration, configuration, assessment, and analysis.

PBSE provides a data model and framework that is both holistic and compact. It addresses the core system science, or first principles of systems required to design complex systems by making interactions more visible and directly relating these to how they deliver value described by stakeholders, noted as features in the S*Metamodel. Additional benefits of the PBSE approach include:

- Strong expression of fitness landscapes as the basis for selection, trades, improvements, decisions, innovations, configuration, and understanding of risk and failure.
- Explication of the system phenomenon (Schindel 2016) as a real world-based science and math foundation for systems engineering, amenable to systems science, connected to historical math/science models of other engineering disciplines, and encouraging discovery and expression.
- A detailed MBSE approach to platform management for system families and product lines.
- Compatibility with contemporary modeling language standards.
- Direct mapping into contemporary modeling tools, PLM information systems, and other COTS tools and enterprise systems, increasing the value of existing information technologies.
- Deeper support for federated data across differing information systems, for integration with emerging open systems life cycle standard technologies.

Pattern-based systems engineering (PBSE) is a methodology which explicates system value through an understanding and explicit modeling of first principles better uniting the systems engineering and decision analysis capabilities. ■

REFERENCES

- Alexander, Christopher. 1964. Notes on the Synthesis of Form. Cambridge, US-MA: Harvard University Press.
- Alexander, Christopher, Sara Ishikawa, Murray Silverstein, Max Jacobson, Ingrid Fiksdahl-King, and Shlomo Angel. 1977. *A Pattern Language*. New York, US-NY: Oxford University Press.
- Bradley, J., M. Hughes, and W. Schindel. 2010. "Optimizing Delivery of Global Pharmaceutical Packaging Solutions, Using Systems Engineering Patterns." Paper presented at the 20th Annual International Symposium of INCOSE, Chicago, US-IL, 11-15 July.
- Cisco. n.d. Internet of Things (IoT) Graphic https://www.ncta.com/platform/industry-news/infographic-the-growth-of-the-internetof-things/.
- Clemen, R. T., and T. Reilly. 2001. *Making Hard Decisions with DecisionTools®*. Pacific Grove, US-CA: Duxbury.
- Cloutier, Robert. 2008. Applicability of Patterns to Architecting Complex Systems: Making Implicit Knowledge Explicit. VDM Verlag Dr. Müller.
- CMU/SEI. 2010. CMMI® for Development (CMMI-DEV) Version 1.3–CMU/SEI-2010-TR-033. Carnegie Mellon University, Software Engineering Institute, Pittsburgh, US-PA, November.
- CESUN. n.d. http://cesun.mit.edu/about/purpose
- Defense Acquisition University. n.d. https://acc.dau.mil/CommunityBrowser.aspx?id=638297.
- Gamma, Erich, Richard Helm, Ralph Johnson, and John Vlissides. 1995. *Design Patterns: Elements of Reusable Object-Oriented Software*. Reading, US-MA: Addison-Wesley Publishing Company.
- ICTT. 2013. Abbreviated Systematica Glossary, Ordered by Concept V 4.2.2. ICTT System Sciences.
- INCOSE. 2007. Systems Engineering Vision 2020–INCOSE-TP-2004-004-02. San Diego, US-CA, September.
- Keeney., R. L. 1992 Value-Focused Thinking — A Path to Creative Decision Making. Cambridge, US-MA: Harvard University Press.
- OMG MBSE. n.d. http://www.omgwiki.org/MBSE/doku.php.
- OMG Patterns Working Group. n.d. http://www.omgwiki.org/MBSE/doku.php?id=mbse:patterns:patterns.
- Schindel, Bill, and Troy Peterson 2013. "Introduction to Pattern-Based Systems Engineering (PBSE): Leveraging MBSE Techniques." Tutorial presented at the INCOSE 2013 Great Lakes Regional Conference on Systems Engineering, October.

- Schindel, W. 2005a. "Pattern-Based Systems Engineering: An Extension of Model-Based SE." Tutorial at the 15th Annual International Symposium of INCOSE, Rochester, US-NY, 10-15 July.
- Schindel, W. 2005b. "Requirements Statements Are Transfer Functions: An Insight from Model-Based Systems Engineering." Paper presented at the 15th Annual International Symposium of INCOSE, Rochester, US-NY, 10-15 July.
- Schindel, W. 2010. "Failure Analysis: Insights from Model-Based Systems Engineering." Paper presented at the 20th Annual International Symposium of INCOSE, Chicago, US-IL, 11-15 July.
- Schindel, W. 2011a. "The Impact of 'Dark Patterns' on Uncertainty: Enhancing Adaptability in the Systems World." Presented at the INCOSE 5th Annual Great Lakes Regional Conference on Systems Engineering, Dearborn, US-MI, 4-6 November.
- Schindel, W. 2011b. "What Is the Smallest Model of a System?" Paper presented at the 21st Annual International Symposium of INCOSE, Denver, US-CO, 20-23 June.
- Schindel, W. 2012. "Integrating Materials, Process & Product Portfolios: Lessons from Pattern-Based Systems Engineering," in Proc. of 2012 Conference of Society for the Advancement of Material and Process Engineering.
- Schindel, W. 2013. "System Interactions: Making the Heart of Systems More Visible." Presented at the INCOSE 7th Annual Great Lakes Regional Conference on Systems Engineering, West Lafayette, US-IN, 5-6 October.
- Schindel, W. 2015. "Pattern Based System Engineering Methodology." MBSE Initiative, Methodology Summary for INCOSE, June.
- Schindel, W. 2016. "Got Phenomena? Science-Based Disciplines for Emerging Systems Challenges." Paper submitted to the 26th Annual International Symposium of INCOSE, Edinburgh, GB-SCT, 18-21 July.
- Schindel, W., and R. Dove. 2016 "Introduction to the Agile Systems Engineering Life Cycle MBSE Pattern." Paper submitted to the 26th Annual International Symposium of INCOSE, Edinburgh, GB-SCT, 18-21 July.
- Schindel, W., and V. Smith, 2002. "Results of Applying a Families-of-Systems Approach to Systems Engineering of Product Line Families." SAE International, Technical Report 2002-01-3086.
- Walden, D., et al., ed. 2015. *The Systems Engineering Handbook: A Guide for System Life Cycle Processes and Activities* Version 4. INCOSE, San Diego, US-CA: Wiley.

ABOUT THE AUTHORS

[Editor: Author biographies were current when the paper was initially published in 2016.]

Troy Peterson is a Booz Allen Fellow and chief engineer at Booz Allen Hamilton. His experience spans commercial, government and academic environments across all product life cycle phases. Troy is INCOSE's assistant director for systems engineering transformation and the co-lead of the Patterns Working Group. Troy received his B.S. in ME from Michigan State University, his M.S. in technology management from RPI, and an advanced graduate certificate in systems design and management from the MIT. He is also INCOSE CSEP, PMI PMP and ASQ SSBB certified.

Bill Schindel is president of ICTT System Sciences. His engineering career began in mil/aero systems with IBM Federal Systems, included faculty service at Rose-Hulman Institute of Technology, and founding of three systems enterprises. Bill co-led a project on systems of innovation in the INCOSE System Science Working Group, co-leads the Patterns Working Group, and is a member of the lead team of the INCOSE agile systems engineering life cycle project.

Author1 and Author2 continued from page 24

- Schindel, W., and R. Dove. 2016. "Introduction to the Agile Systems Engineering Life Cycle MBSE Pattern." Paper presented at the 26th Annual International Symposium of INCOSE, Edinburgh, GB-SCT, 18-21 July.
- Schindel, W. 2007. "Improving Design Review." ICTT System Sciences.
- Servos, John W. 1996. *Physical Chemistry from Ostwald to Pauling*, Reprint Edition, Princeton, US-NJ: Princeton University Press.
- Sussman, G., and J. Wisdom. 2001. *Structure and Interpretation of Classical Mechanics*. Cambridge, US-MA: MIT Press.
- Walden, D., et al., eds. 2015. *Systems Engineering Handbook: A Guide for System Life Cycle Processes and Activities* Fourth Edition. INCOSE, San Diego, US-CA: Wiley.
- Warfield, John N. 2006. *An Introduction to Systems Science*. Hackensack, US-NJ: World Scientific Publ.
- Westfall, Richard S. 1980. *Never at Rest: A Biography of Isaac Newton*. Cambridge, GB: Cambridge University Press.
- Wymore, A. Wayne. 1967. *A Mathematical Theory of Systems Engineering: The Elements*. New York, US-NY: Wiley.
- Wymore, A. Wayne. 1993. *Model-Based Systems Engineering*. Boca Raton, US-FL: CRC Press.
- "Gene regulatory network." https://en.wikipedia.org/wiki/Gene_regulatory_network.
- ICTT. 2007. "Improving Design Review" V1.4.1. ICTT System Sciences.
- INCOSE PBSE Working Group. 2015. "Pattern-Based Systems Engineering (PBSE), Based On S*MBSE Models." http://www.omgwiki.org/MBSE/doku.php?id=mbse:patterns:patterns_challenge_team_mtg_0 6.16.15.

ABOUT THE AUTHOR

[Editor: Author biography was current when the paper was initially published in 2015.]

William D. (Bill) Schindel is president of ICTT System Sciences. His engineering career began in mil/aero systems with IBM Federal Systems, included faculty service at Rose-Hulman Institute of Technology, and founding of three systems enterprises. Bill co-led a 2013 project on the science of systems of innovation in the INCOSE System Science Working Group. He co-leads the patterns challenge team of the OMG/INCOSE MBSE initiative, and is a member of the lead team of the INCOSE agile systems engineering life cycle discovery project.

Innovation, Risk, Agility, and Learning, Viewed as Optimal Control and Estimation

William D. (Bill) Schindel, schindel@ictt.com

[Editor: This paper for systems engineering foundations refers to the *Systems Engineering Handbook* 4th edition (Copyright 2015 by the International Council on Systems Engineering), ISO 15288:2015, and the Systems Engineering Vision 2020 published by INCOSE in 2007.]

■ **ABSTRACT**

This paper summarizes how a well-understood problem—optimal control and estimation in "noisy" environments—provides a framework to advance understanding of a well-known but less well-mastered problem—system innovation life cycles and management of decision risks and learning. The ISO15288 process framework and its exposition in the INCOSE *Systems Engineering Handbook* (2015) describe system development and other life cycle processes. Concerns about improving the performance of processes in dynamic, uncertain, and changing environments are partly addressed by "agile" systems engineering approaches. Both are typically described in the procedural language of business processes, so it is not always clear whether the different approaches are fundamentally at odds, or just different sides of the same coin. Describing the target system, its environment, and the life cycle management processes using models of dynamical systems allows us to apply earlier technical tools, such as the theory of optimal control in noisy environments, to emerging innovation methods.

INTRODUCTION: OVERVIEW AND BACKGROUND

Problem Statement

Advancing understanding and performance of the system innovation life cycle is central to INCOSE. Current understanding is exemplified by the *Systems Engineering Handbook*, ISO15288, and Guide to the SE Body of Knowledge (Walden et al. 2015, ISO 15288:2015, and Pyster et al. 2013), describing established principles and practices grown pragmatically out of decades of real-world experience. This is a different kind of foundation than STEM understanding of the phenomena of electrical, mechanical, or chemical systems as the basis for electrical, mechanical, and chemical engineering disciplines. Complex engineered systems and environments, systems of systems, compressed innovation cycles, and dynamically changing competitive markets and technologies challenge understanding and capabilities to perform system innovation effectively enough. The traditional principles might still apply, but how do we know whether we are performing the overall process as effectively as possible? We understand the possibilities and limits on efficiency of engines from thermodynamics, but how do we understand the possibilities and limits on innovation cycles? This paper suggests that certain existing STEM-based foundations are available, enabled by the transition to model-based systems engineering (MBSE), that can be exploited in pursuit of optimizing innovation cycles, and as a foundation for understanding currently emerging methods.

The Geometrization of Innovation Space

Converting physical and mathematical descriptions into "spatial" geometric terms, while seemingly abstract, has a long history of positive impacts in the history of science, technology, engineering, and mathematics (STEM). This paper introduces the same kind of thinking into how we understand the process of innovation in general as a system, and particularly in more challenging cases involving highly dynamic environments, continuous learning, and uncertainties in our ability to fully observe

or control what is occurring during the innovation process. Even where current practices may be seen in this approach, it provides a more general way to understand them, and therefore to perform and improve them in the future.

Before introducing this alternate perspective, this paper will briefly summarize some of the traditional as well as emerging perspectives across seemingly different domains. Then, the impact of shifting to more model-based representation of systems on our ability use the ideas of mathematical spaces will be described. Following that, the paper will describe the spaces of interest in this case, and how they are addressed by the existing theory of optimal control and estimation. An immediate application is noted in the world of agile, "continuous, and "fail fast and recover early" development, and other applications are briefly summarized, with additional application suggestions for future pursuit.

Innovation, Risk, and Agility: Traditional and Emerging Concerns

This section briefly summarizes some of the more prominent risk-connected aspects of traditional and emerging perspectives on system innovation and related life cycle management. For purposes of this discussion, we will consider innovation to mean the delivery of improved stakeholder value, through any aspects of the system life cycle management processes. This is an explicit formalism, because the approach explicitly models value across all stakeholders (Kline et al. 2017, Simoni et al. 2016, and Rogers 2003). It avoids a technology-centric view, without ducking the challenge of complexity. It also creates an explicit space for improved understanding of variation and selection.

Concerns of Traditional Approaches to Innovation. The traditional systems engineering view of these life cycle processes can be described by the ISO15288 standard (ISO 2015) or its further description in the INCOSE *Systems Engineering Handbook* (Walden et al. 2015). Within that perspective, a number of differently configured specific forms of the development portion of these life cycles may apply, based on the metaphors of waterfall, spiral, waves, or otherwise. Additional portions of the traditional ISO 15288 life cycle processes include production, distribution, operation, maintenance, update, and retirement, any of which may be subject to innovation delivering enhanced stakeholder value.

The risk management perspective in the traditional case would include concerns such as multiple types and sources of risk, among these limited knowledge of changing environment, stakeholder situations and needs, as well as technical and other risks to performance, costs, and schedule. Traditional risk management concerns include identifying risks, assessing them, and working to avoid, transfer, mitigate, and monitor those risks (Walden et al. 2015). More attention is recently paid to risks arising when systems of interest or their environments exhibit dynamical complexity (Sheard et al. 2016).

Concerns of Emerging Approaches to Innovation. New approaches to innovation are rapidly emerging and are sometimes perceived (correctly or not) as at odds with systems engineering, at least as traditionally performed. In the agile and lean start up communities (for systems, software, products, business, etc.), risk is addressed by seeking early and continuing feedback through short or incremental "experiments" (whether called "sprints" or otherwise) that encourage discovery, exposure, or exploration of instances of risk early enough that they can be addressed while the cost of doing so is still relatively smaller, even if this causes change to what would otherwise have been fundamental assumptions. Examples can be seen in the methodologies of agile software and systems (Rigby et al. 2016, Dove and LaBarge 2014), lean start up, the minimum viable product, pivoting (Ries 2011), and experimentation in general (Schrage 2014, Anderson et al. 2011, Clarke 2016, Kohavi et al. 2009, Manzi 2012, Teller 2016, and Thomke 2003).

How System Models Can Shift Our Perspective on Innovation

INCOSE has recognized the importance of the continuing rise of model-based methods (Friedenthal et al. 2015), and formalized an objective of supporting systems engineering becoming a model-based discipline (Peterson et al. 2017). We note that this emergence is still at a relatively early and progressing stage—what is currently referred to as a "system model" may not represent what is possible in the future. This larger shift can include moving from system engineering's traditionally process- and procedure-oriented emphasis to something closer to the system model emphasis of other STEM disciplines (Schindel 2016), without abandoning the discipline of process.

Mathematically oriented models have a long history in design optimization, (Fisher 1971, Bellman 1957, Koch 1998, Pontryagin et al. 1962, Smaling 2005, and Box 2013), predating more recent use of system (MBSE) models for other purposes (Friedenthal et al. 2015). However, the scope of such design optimization mathematical models was generally focused on key architectural or other specific, important, but limited scope decisions, not the overall system being innovated, and not a dynamical model of the overall process of innovation.

In contrast to but building on that history, our interest in this paper is the convergence of (1) the earlier design optimization models (cited above) with (2) wider-scope, system-level MBSE models having strengthened STEM foundations (cited above), (3) more powerful computational environments (Friedenthal et al. 2015), (4) continuous incremental development methodologies (cited above), and (5) extension of the system models to include both the target system of interest and the development and other life cycle management environments as systems in their own right (Schindel and Dove 2016).

The traditional issues summarized in the earlier sections above are fundamental and not likely to disappear through technique or method. However, the rise of system models as tools for innovation can have similar effects to their historical emergence in the other scientific and engineering disciplines—increasingly powerful ways to understand and attack those traditional issues, with increased clarity, quantification, and qualitative understanding.

WHERE DO SYSTEMS COME FROM? SYSTEM LIFE CYCLE TRAJECTORIES IN S*SPACE

SE Information versus SE Process

The systems engineering process is often conceptualized by systems engineers using the life cycle management process models of ISO 15288 and the INCOSE *Systems Engineering Handbook*, exemplified by the systems engineering "vee" model (Forsberg et al. 2000), in one form or another, such as illustrated by the upper portion of Figure 1.

As also illustrated in Figure 1, *the systems engineering process consumes and produces information*, along with other kinds of resources. The perspective of this paper assumes an INCOSE-visualized future of model-represented information, representing system configurations progressing over the system life cycle. Because this paper emphasizes the impact of system models, Figure 1 uses symbology from the S*Metamodel summary framework (INCOSE Patterns Working Group 2015) to illustrate the iterative production and consumption of information within the systems engineering process. The S*Metamodel framework represents the smallest set of information sufficient for the purposes of science and engineering in model form including: a significant range of stakeholder value/fitness space and purpose, technical requirements, design architecture, quantitative couplings and sensitivities, and failure modes and impacts.

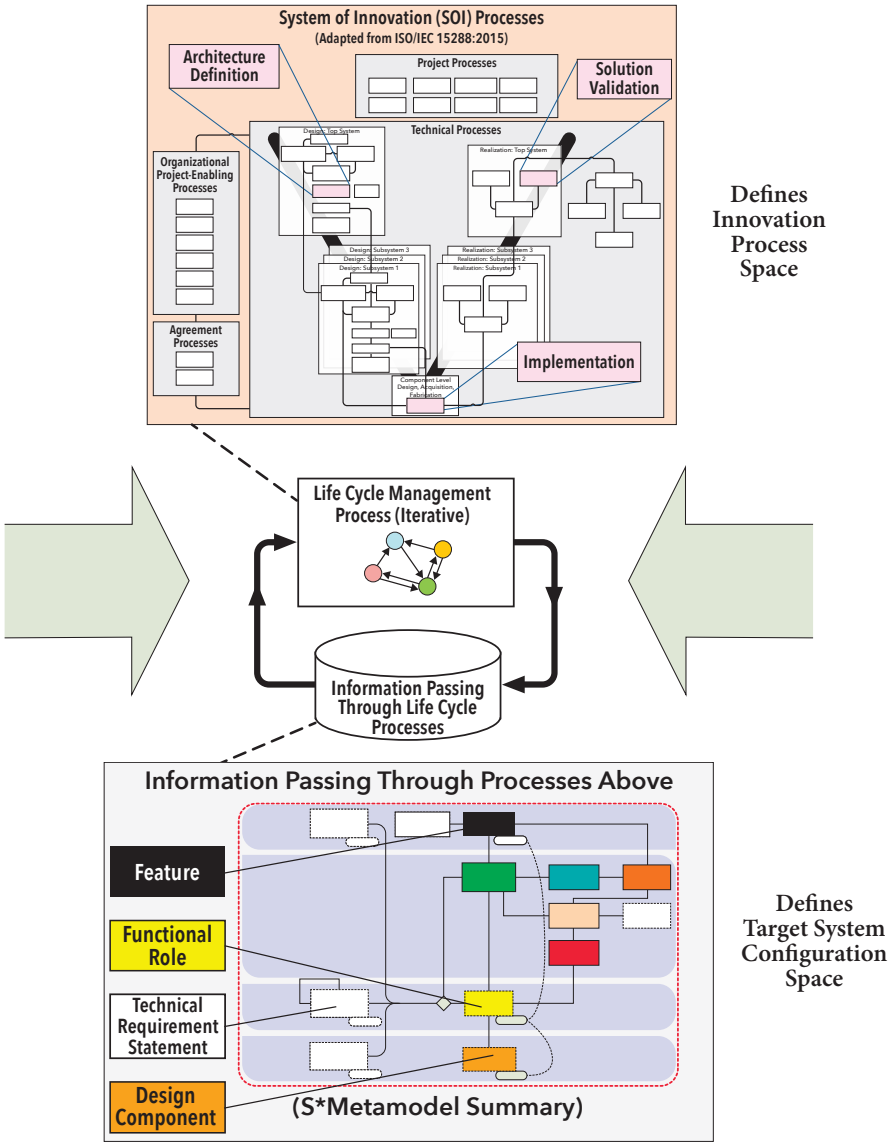


Figure 1. The systems engineering process produces and consumes information

Traditional systems engineering has historically emphasized process and procedure over the information those processes consume and produce. As evidence (Walden et al. 2015, Schindel 2015, and INCOSE MBSE Patterns Working Group 2015) of this relative emphasis, one may refer to the amount of ink and paper spent to describe expected process and procedure versus to describe the expected information consumed and produced. The referenced industry and enterprise process standards certainly refer to both process as well as information, but the rise of model-based methods is shifting the relative balance of these two back in the direction of information models. It is informative to compare this to the history of physical science-based engineering disciplines (ME, CE, ChE, EE, etc.), in which there is relatively more emphasis on the models of underlying phenomena and system models, and relatively less emphasis on the "procedure for performing electrical engineering". As noted in (Schindel 2005, 2015, 2016), historical impacts of this situation have included:

1. Difficulty determining when we are "done" performing systems engineering, measuring where we are in process and procedure space (top of Figure 1) instead of where things stand in modeled target system configuration space (bottom of Figure 1).
2. In the same way, more subjectivity than would be desired in describing what comes next, by referring to procedural steps (procedure space) instead of modeled target system configuration space progress.
3. Ambiguity in what the procedure-oriented approach says we should do with what we already know from past projects, versus what we are finding out for the first time. (Have we ever designed a power supply before?)
4. More subjectivity and interpretation are required in reviews than would be preferred.
5. Arguments about whether systems engineering has its foundation, like the other engineering disciplines, in underlying phenomena, physical laws, and first principles.

Geometrization of Systems Engineering Model Information Space

By "geometrization," we refer to the use of spatial coordinate system frameworks to represent state (in this case, the configuration of a modeled system), and with that transformation the availability of certain important formal mathematical tools coupled with intuitive spatial references. A familiar framework of this sort is a three-dimensional representation of space above a small region of the surface of the earth, used to represent the ballistic trajectory of a projectile fired from and returning to the earth. Other geometrized representations describe more abstract ideas in more familiar looking 3-space, or higher numbers of dimensions.

Two very famous geometrizations occurred in the history of mathematics, both having profound practical impacts on the day-to-day tools of modern engineering, noted in Figure 2.

1. The geometrization of algebra, by Rene Descartes, associated with graphs of conic sections or other shapes generated by algebraic formulae (Boyer 1959).
2. The geometrization of function

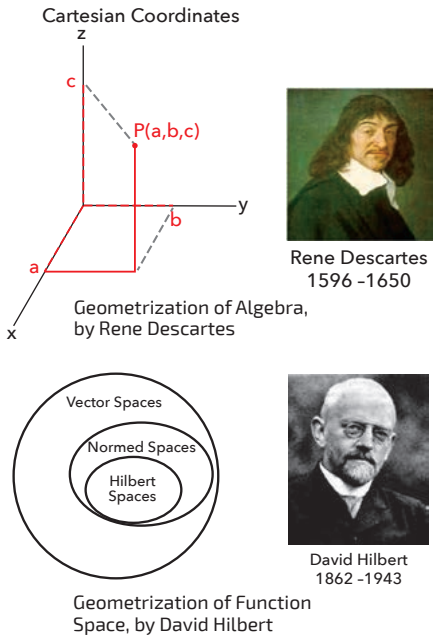


Figure 2. Two geometrizations had enormous impact: Descartes and Hilbert

spaces, by David Hilbert, associated with function inner products and distance metrics, correlations, angular direction, frequency domain transformations and projections, etc. (Simmons, 2003).

The practical effect of these become available when we begin to describe innovated systems using models, if the models are based on a strong enough metamodel foundation:

A. Viewing the configuration of system information (whether about stakeholder value and system fitness, or technical requirements, or design architecture, or failure modes, or sensitivities and couplings, or otherwise) as a point in system configuration space.
B. Visualizing "where we are" in an innovation process as a (moving) point in that system configuration space, representing the current understanding of the system of interest—instead which step of a procedure we have completed. We begin to think in system configuration space instead of process and procedures pace. (See Figure 1.)
C. Visualizing "where we are going" as points we want to reach in system configuration space, instead of steps in process and procedure space.
D. Taking advantage of the mathematical concepts and tools that go with such spaces, including distance metrics, velocities, inner products, projections, and other tools.
E. Visualizing the progression of points in system configuration space as a trajectory, along which we want to move during innovation in an optimal way toward a goal.
F. Realizing that this has converted the problem of innovation into one of optimal dynamical travel along an (agile) trajectory, in the presence of uncertainty.

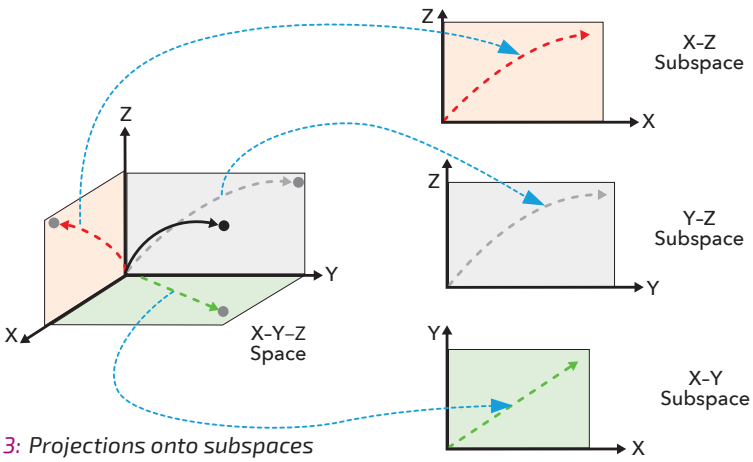


Figure 3: Projections onto subspaces

Nothing about the above should be interpreted as suggesting that innovation is a simple deterministic process (quite the opposite—many random processes are involved), or that we can predict its outcomes (also not so), fail to use the deep lessons learned by experienced leaders in traditional environments (rather, we want to share them more widely), or that we are not including serendipity or creativity (think about models of biological innovation or earthquakes). Rather, we are suggesting that, as with other applications of science and engineering, we are seeking STEM models of the world we occupy, to improve our ability to learn and succeed within it.

*Trajectory Projections in S*Subspaces*

The full S*Configuration space set of information modeled across the engineering process and system life cycle has much higher dimension than three-space, and involves a mixture of dissimilar ideas, such as stakeholder value, material properties, laws of physics, etc. That may suggest putting all this into an integrated configuration space is too daunting a task.

However, a powerful aspect of geometrization is the idea of subspaces, in which some dimensions are temporarily ignored and a smaller number of current interests are visualized. This idea is illustrated by Figure 3, in which a trajectory in 3-space is projected onto three different sub-spaces, each of two dimensions.

In the same way, subsets of the S*Configuration space may be separately studied, for a system that has projections into many subspaces. Figure 4 shows three such subspaces of interest, each of which represents potential creative or discovered syntheses:

1. **Stakeholder feature subspace:** A discovered or learned synthesis of stakeholder types and their respective value or fitness space, against which systems will be judged. The place where the value of delivered innovation is ultimately realized and validated.
2. **Technical behavior subspace:** A

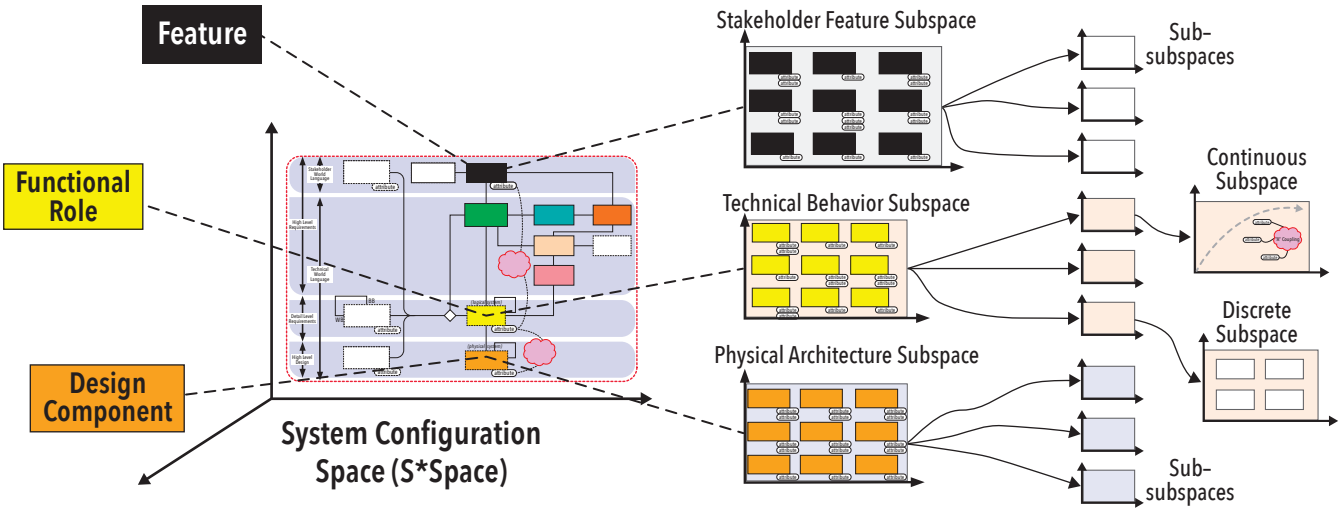


Figure 4: Stakeholder feature subspace; technical behavior subspace; design subspace

discovered or learned concept of operations and its related black box technical specification. The place where a technical behavior appears as a potential way to deliver value in (item 1 just above), and where a candidate's design performance is judged technically.

3. **Physical architecture subspace:** A discovered or learned design solution, including physical architecture, the technologies upon which it is constructed, and the means of delivering the technical performance called for in (item 2 just above).

This is not to suggest that projection onto subspaces is something new: model views, reducing dimensions, applying principal component analysis, and the like are familiar enough in engineering. Rather, what we are pointing out is that MBSE is nearing the point at which the whole system innovation problem — not just a part of it — can be cast in this framework. At this point, the "guidance system" discussed in the next section becomes a practicality, as follows.

A system's configurations, during the innovation cycle, can now be conceived as moving along a trajectory in each of those individual subspaces, representing projections onto each of them, from the combined trajectory in the total space. We can consider paths that are more or less desirable, think about velocity along the path, ideas of uncertainty about location, development response time, agility, and other important issues. The idea of optimality of trajectory now becomes more clearly related to innovation over life cycles. This optimality may have to do with minimizing transit time, response or recovery time, resources expended along the trajectory, and uncertainty as to position or other feedback.

The scope of S*Configuration space thus includes not just issues of technical requirements and design, but also identity of stakeholders and models of stakeholder value. This means that innovation opportunity is a part of this space, and the innovation process includes discovery of opportunity and purpose, not just design. We are reminded that this trajectory includes discovery and learning about all three of the subspaces in Figure 4, and others, bearing on current interest in emergence or discovery of purpose, and "pivoting" (Reis 2011).

This nearly brings us to the point of having transformed the view of innovation to a view of optimal trajectory guidance in a noisy environment, but we still need to add the guidance system, as well as arrangements for learning.

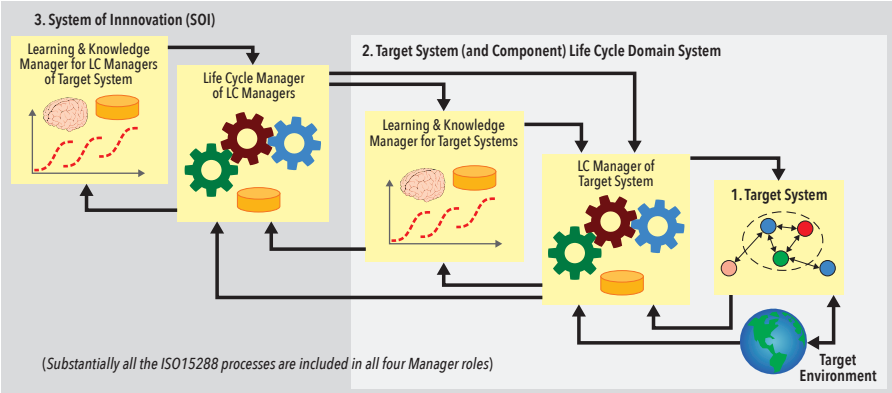


Figure 5. The ASELCM pattern: top level reference boundaries

The Guidance System: Including the System of Innovation in the Model

Based on the above, we now have the target system, subject to innovation, represented by a model, having a configuration to be guided along an innovation trajectory path in system configuration Space. The "guidance system" for that trajectory becomes the life cycle management systems of ISO 15288, including systems engineering and other processes. We are still very interested in that ISO 15288 process set, which has great community-learned reference value, but we can also view it in a new light, as follows.

The traditional "vee diagram" view of the ISO15288 model (upper part of Figure 1) focuses on key interdependencies of the life cycle management processes, arising from the nature of developed systems, and with an emphasis on the management of those processes. What we will see below emphasizes different aspects of the same processes — the discovery, learning, and use of learning aspects, and how they relate to the very same ISO15288 processes. It is a different emphasis on the traditional processes — not an abandonment of them.

A reference model is shown in Figure 5, the agile systems engineering life cycle management (ASELCM) pattern, in use by the INCOSE ASELCM discovery project (Schindel and Dove 2016). It includes three major subsystems:

1. **System 1:** Target system of interest, to be engineered or improved. (The system modeled in the earlier sections above, whose configuration trajectory is to be guided.)
2. **System 2:** The environment of (interacting with) System 1, including not only its operational environment, but also all the life cycle management systems of System 1, *including learning about System 1 and its environment*.
3. **System 3:** The life cycle management systems for System 2, including learning about Systems 2 and its environment.

Note that System 2 is further divided into:

A. **Learning and knowledge manager for target system:** Discovers and accumulates new and existing knowledge about System 1 and its operating environment.
B. **Life cycle manager for target system:** Uses what has already been learned (in A above) about System 1, performing all the necessary life cycle management processes.

The same sort of sub-division occurs for System 3 but concerned with discovery and learning about System 2 and its environment, and managing its life cycle. So, System 3 includes all process improvement for System 2.

The ASELCM pattern of Figure 5 shows observation and feedback loops. This pattern models I nnovation itself, not just the innovated thing—and is non-linear, iterated, and exploratory as to configuration space. It is a complex adaptive system reference model for system innovation, adaptation, operation/use/metabolism, sustainment, and retirement or replacement. It applies to 100% human-performed or automation-aided innovation, or hybrids thereof, whether performed with agility or not, ISO 15288 oriented or informal, and whether performed well or poorly. It includes representation of pro-active, anticipatory systems. The rise of a number of newer innovation methods and emphases, in business and technical systems, supports the need for such a combined reference model:

1. Agile engineering of systems and software (Dove and Labarge 2014, Rigby et al. 2016)
2. Product line engineering of composable, configurable systems (INCOSE PLE WG 2015)
3. Experiment-based innovation (Schrage 2014, Anderson et al. 2011, Manzi 2012)

4. Fail fast and recover early (Dove et al. 2016)
5. Lean business start up, the minimum viable product, and pivoting (Ries 2011).

Effective Learning: More than "Lessons Learned" Reports

The emerging innovation methods cited above particularly emphasize learning, whether it is discoveries about stakeholders and their value space, the evolving environment, competitive alternatives, system concept of operations and technical requirements, designs and technological characterization, or failure modes and design limits. As methodologies couched in agility, experiment, pivot, or fail fast and recover early, the hallmark of these methods is admission that a changing or uncertain world creates risks and opportunities in the form of incomplete knowledge. Of course, this has always been true, at least to some degree, in the world of innovation, traditional or otherwise—but the newer methods particularly emphasize means of accelerating the related discovery and learning process, managing related risks.

Accordingly, strategies for learning are of particular importance (Christensen et al. 2011, Schindel et al. 2011). This learning amounts to filling in more knowledge in the models of the configuration spaces described above. Because these spaces are usually very large, with many degrees of freedom and parametric ranges, and because exploration, experimentation, and learning require expending time and other resources, the strategy for picking what to learn about, what to invest experiment and learning resources in, becomes important. The concept of configuration space and trajectories through it can help us see this exploration as "flying through" the space in designated "search patterns". Interest in optimal strategies (that is, trajectories, routes) for exploration of this space becomes a natural extension of the theory of design of experiments (Fisher 1971), and has become the subject of a significant literature on experiment, in its own right (Schrage 2014, Anderson et al. 2011, Clarke 2016, Kohavi et al. 2009, Manzi 2012, Teller 2016, and Thomke 2003).

For the systems engineering process, there are a number of learning-related implications:

1. **How is continuous, incremental learning represented?** In the approach described above, what is already known about System 1 is represented by the smallest model sufficient for purposes of engineering or science. It follows that what is learned in the future about System 1 would be represented as (incremental) changes to that model.
2. **Learning must be compressed and placed "in the way" of future performance:** For learning to be effective, it must impact future behavior. Just "storing" what is learned is not the objective, which is improved future performance about what was learned. So, what was learned must be effectively incorporated in future performance. While the internal means of this are somewhat masked by biology for individual humans, when it comes to teams and enterprises, we must ask how learning is to improve future performance of the group. We suggest that it is not effective to accumulate ever-growing sets of "lessons learned reports", even if searchable as databases. The INCOSE MBSE Patterns Working Group describes S*Patterns as the configurable, re-usable models of whole target systems (INCOSE Patterns Working Group 2015). These are subsequently configured as the starting point of future performance, so that whatever has flowed into the patterns becomes a (configurable, as needed) part of future performance. Think of "muscle memory" in humans.
3. **Learning in each ISO 15288 process:** Figure 6 shows that the ISO 15288 life cycle management processes appear twice in System 2 and twice in System 3. Two of those appearances are learning processes—they are the learning aspect of each of the (already defined) ISO 15288 processes. They are about learning new things about the subject of those processes—whether they are about stakeholder or technical needs, designs, verifications, or otherwise. Every ISO 105288 process potentially has a learning aspect. But each of them also has a "non-learning" execution

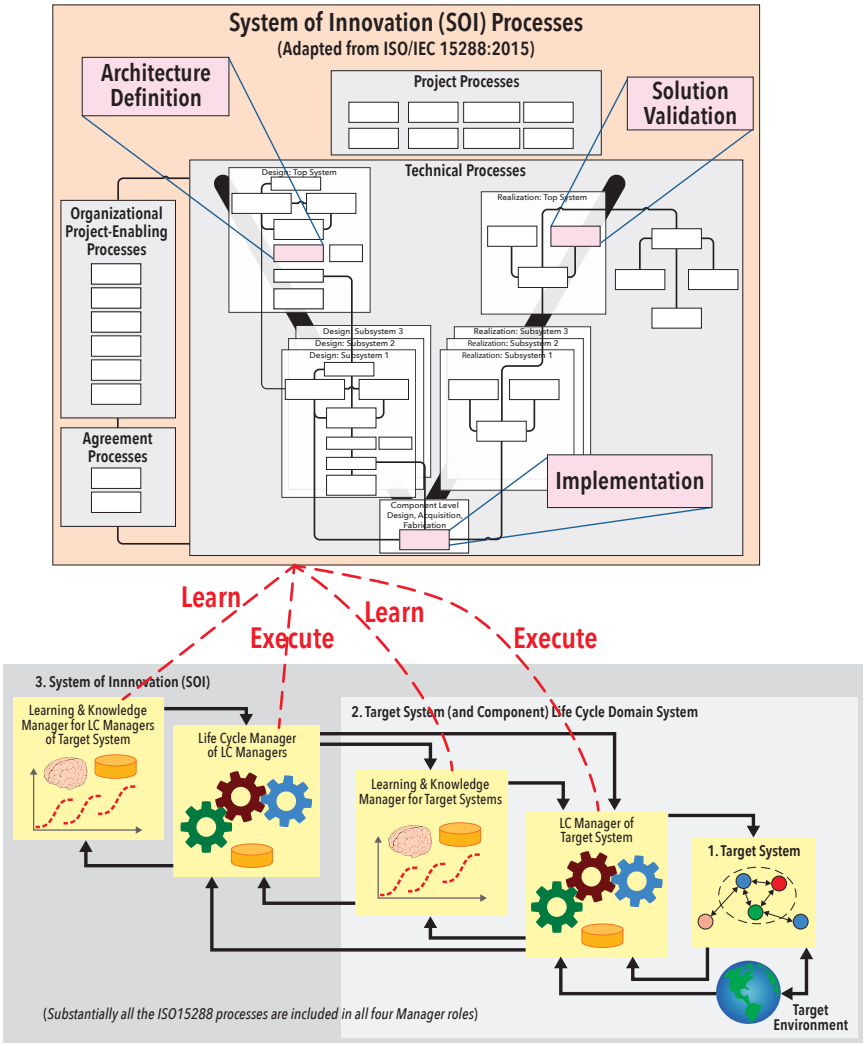


Figure 6. The systems engineering "vee" appears four times in the ASELCM pattern

only aspect, in which what has already been learned is applied. It is not the case that engineering a system requires learning. In the case of product line engineering (PLE) for configurable platforms, there are rapid-execution versions of each of the ISO 15288 processes that essentially "configure" what is already defined in the platform pattern, for a specific case. The platform and its supporting patterns represent what was learned in the past—what we already know.

4. **What about what we already know?** The traditional description of the systems engineering process actually describes all the things we would do if we knew nothing in advance about a system or its domain. But what about what we already know, which is usually quite a lot? Very little of the traditional life cycle process description addresses that question, nor how it would be blended with new learning processes. So, splitting up processes into the learning – execution pairs of Figure 5 have the further advantage of explicating this important aspect, essential to agility.
5. **Learning about System 2:** These same points, concerning System 2's learning about System 1, will also apply to System 3's learning about System 2.

WHAT OPTIMAL CONTROL AND ESTIMATION THEORY CAN TELL US

It is hard to overstate the transformative successes and spread, during the last fifty years, of the theory of optimal estimation, with related technologies for extracting signals from noisy environments, and the theory of optimal control, with applications of feedback control systems. Among the key modern contributors to these underpinnings have been Norbert Wiener (time series, fire control systems, feedback control, cybernetics), Rudolph Kalman (filtering theory, optimal Bayesian estimation), Lev Pontryagin (optimal control, maximum principle), and Richard Bellman (dynamic programming). Applications spread from defense fire control systems, through multi-sensor navigation systems, to control strategies implemented in manufacturing, transportation, energy, communication, medical, entertainment, scientific instrumentation, and other domains. Without these accomplishments, much of modern life would disappear or shift to much less favorable human experience of a century or more earlier (Wiener 1949; Kalman 1960; Pontryagin et al. 1962; Bellman 1957, 1959; Bryson 1967; Bryson and Ho 1975).

These successes have been powered by mathematical models of the related (engineered) systems of interest and their environments. These include their equations of motion (state) and model elements representing measurement, control, uncertainty, risk, and feedback. The accumulation of progress in the capabilities of related models and technologies stands in contrast to the progress of the less formal theories and practice of human organizations, business and management, including the process and procedure of systems engineering in particular, or human-performed innovation in general. While these latter human activities have clearly progressed in very important ways, they have been supported by less formal descriptions, subjective judgement, and human intuition—all of these powerful but something different than the above-referenced theory and applications of optimal control and estimation. A reasonable and expected first reaction would be that they simply do not apply in a concrete way, because it has simply not been clear how to practically apply those tools to complex problems such as the management of development processes. So, the latter have been described by informal prose, including prominent examples such as the INCOSE Systems Engineering Handbook and the ISO 15288 life cycle management standard.

Is It Plausible To Apply Optimal Control to the Innovation Process?

As the underlying approaches to model-based representations of systems are progressing, we may ask whether this progress is yet sufficient to help us to apply the more powerful mathematical frameworks to the domain of innovation itself. Is it plausible that optimal control and estimation might have practical application to the innovation process itself? And, if it is, why has this not occurred on a widespread basis already?

We first review the nature of these technical frameworks, as they have succeeded in their contemporary application domains, then ask whether and how they apply to innovation itself.

Optimal estimation theory is based on models of estimation, from noisy (corrupted) observations or measurements, of the current state of an (also modeled) system, which may itself also be driven by random processes. This framework addresses the question of how to optimize those estimates, as to their uncertainty. Optimal control theory begins with models of a system's equations of motion, including the model of its environment and drivers, adds models of control inputs, and asks how to optimize those control inputs so as to optimize various objective functions, such as trajectory, elapsed response time, frequency response, expenditure of fuel, energy, or other resources, proximity to moving targets or set points, or other more complex objective functions. The deterministic theory is then extended by adding random processes to both system environmental drivers as well as noise-corrupted observation processes. Optimal control objectives are then extended to include uncertainty.

So, how well does the innovation process itself sound like it might fit what the theory of optimal control and estimation addresses? Table 1 (on the next page) compares the application of the theory, applied to a guidance system, to the same theory, applied to a system of innovation. How are the seemingly different concepts of Table 1's middle and right columns in fact similar? The answer is that they play the common roles listed in Table 1's left column. This is similar to the idea that a control system embedded in an automobile and embedded in a manufacturing system still depend upon the same theoretical foundations from controls theory.

The inspiration of vehicle trajectory control as a trajectory metaphor for travel through innovation state space is further supported by the vehicle work of (King et al. 2016 and Martinovich 1988). The typical formulation of the Table 1 left column concepts, independent of domain, is in the next section.

Risk-Optimal Control and Estimation: Typical Problem Frameworks

Mathematical frameworks of optimal estimation, prediction, and control problems, including deterministic and stochastic, linear and non-linear, continuous and discrete time, as well as combinatorial, have been the subject of extensive attention for decades, resulting in many feedback-based applications in estimation and control. While not every class of problem is covered by these advances, their range of successes is formidable.

For comparison to Table 1, a typical time continuous problem statement framework (discrete forms also available) is as follows (Levi 2014, Bryson and Ho 1967):

System defined by: $\dot{\boldsymbol{X}} = \boldsymbol{f}(\boldsymbol{X}, \boldsymbol{U}) + \boldsymbol{W}$, having system state $\mathbf{X}(t) \in \boldsymbol{R}_n$, with control $\mathbf{U}(t)$, driven by process $\boldsymbol{W}(t)$; allowing observations $\boldsymbol{Y} = \boldsymbol{h}(\boldsymbol{X}) + \boldsymbol{V}$, $\boldsymbol{Y} \in \boldsymbol{R}_n$ having observation corruption by random process $\mathbf{V}(t)$.

Find an optimal control U(t) minimizing expected objective functional: $\int_0^T \boldsymbol{g}(\boldsymbol{X}(\boldsymbol{t}), \boldsymbol{U}(\boldsymbol{t}))\boldsymbol{dt}$, and describe the means of quantifying uncertainty based on model and observation.

Table 1. Informal comparison of two domains, as a plausibility test

Aspect of Common Theoretical Framework	Application to a Vehicle Guidance System	Application to a System of Innovation
Overall domain system	Propelled airborne vehicle guidance to moving airborne target	Development of new system configuration for a system of interest
The controlled system	Airborne Pursuit Vehicle	The development process
Control system	Flight control system and pilot sometimes	Development management & decision-making process
Other actors	Target, atmosphere	Stakeholders, operating environment of system of interest, suppliers
State space in which controlled performance occurs	Vehicle position in 3-D geometric space	Configuration space of system of interest, including its features, technical requirements, and physical architecture
Driving processes	Target dynamics, pursuit thrust, flight control surface movements	Stakeholder interest, supply chain
Random aspects of driving processes	Buffeting winds	Stakeholder preferences, competition, technologies
Observation process model	Radar tracking of moving target, sensor characterization	Status reporting, market feedback, development status report process
Random disturbances of observation processes	Sensor errors	Inaccuracies or unknowables in development status; sampling errors
Environmental Conditions	Target maneuvers; atmospheric effects	Market or other environmental conditions;
Control input	Flight control surface orientation	Management direction; resources
Objective function to optimize	Time to target	Time to market; Competitive Response Time; Innovated System Performance; Innovation Risk vs. Reward
Dynamical model	Ballistic Flight, Atmospheric Effects, Thrust	Coupled development processes
Outcome risk	Risk of missing airborne target	Risk of innovation outcomes across stakeholders

In linear, or linearized, cases and for discrete time cases, Figure 7 illustrates the form of a representative feedback system, adapted from Bryson and Ho (1967), where the coefficients shown are generated from system specifications or learning about the random processes from observation (Schindel 1972); other aspects quantify how uncertainty propagates.

The framework and Figure 7 are suggestive, not meant to establish the specific form for the innovation problem summarized in Table 1. However, the annotations added to Figure 7 are practical reminders, even in the most non-linear, manual human-performed control, of more fundamental aspects of management and estimation in uncertain environments, concerning:

1. Use of knowledge of managed system dynamics to predict future state ("dead reckoning" based on beliefs about prior state and system behavior)
2. Use of observational data to correct what was otherwise believed
3. Relative weighting of (1) versus (2)
4. Steering to desired trajectory goals based on current estimated state, goal, and beliefs about system response dynamics
5. Exploration to improve knowledge/beliefs of system structure, dynamics, stochastics.

Just as these ideas are important in any manually human-managed innovation, so they can also be important in applying optimal estimation and control to innovation.

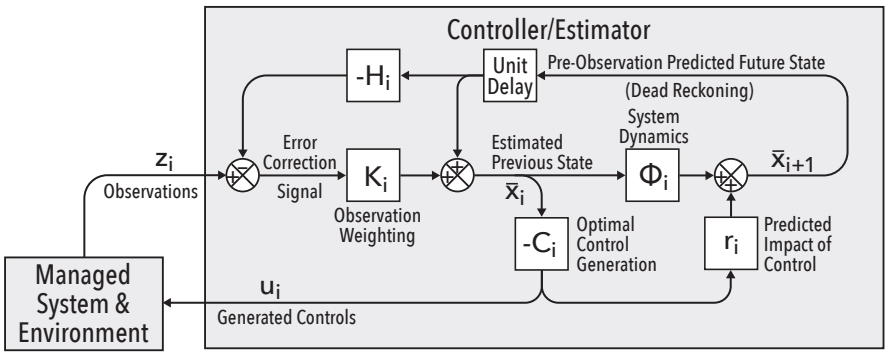


Figure 7. Form of typical optimal stochastic estimator/controller, in linearized discrete time form (adapted from Bryson and Ho 1967 and Schindel 1972)

*Agility as Risk-Optimized Control of Trajectory in S*Space*

Learning trajectories versus mission trajectories. In a dynamic and uncertain environment, the above can help us understand how to plan trajectories that are optimal with respect to two different goals:

1. **Mission response to environment:** Adjusting course (system configuration) in a responsive way, to perform the base mission, or to improve ability to perform the mission. This form of agility exploits what is already known (expressed in the model), based on basic mission and the current or projected operational environment.
2. **Exploration for learning:** This goal is concerned with exploring to capture additional information to improve understanding about (the model of) the system of interest or its environment, and possibly in the presence of random process corruption of observations as well as random processes driving the systems.

(Simkins et al. 2008) illustrates optimal control in a mixed exploitation-exploration approach.

Support for experiment selection in "fail fast and recover early" risk strategy. When dealing with "moon shot" or less familiar areas (for example, early stage technologies, early stage market concepts), concerns of later stage "too late" discovery of infeasibility, financial, or stakeholder issues is significant. The literature on "fail fast and recover early" innovation suggests the strategy of addressing the apparent highest risk issues earliest, to eliminate as soon as possible what turn out to be infeasible choices (Teller 2016). This is a much different strategy than the WSJF (weighted shortest job first) strategy sometimes applied in agile systems engineering to pick next increments (Reinertsen 2009).

Gradient-based versus exploratory direction. Given a current location in S*Space, the principle of optimality (Pontryagin et al. 1962) describes the direction of the optimal trajectory from that point, assuming reachability from that point. If reachability is not assured, then "fail fast" experiments such as in the above approach are suggested.

Intermediate gain delivery trajectories. Even in the case of starting toward a known reachable point, though, agile principles suggest that the trajectory needs to deliver intermediate progress along its route to a destination. That is, intermediate points along the trajectory need to be sought out as intermediate "agile" deliverable configurations that offer incremental improvement in their own right, if the objectives require.

Innovation in Populations: Markets, Segments, Ecosystems

This approach can also be extended beyond trajectories of a single system, by considering populations of systems. In market or ecological frameworks, systems of different configurations of multiply instantiated (populated) instances interact with other systems in roles acting as predators, prey, commercial or military competitors, customers, suppliers, infrastructure, or others.

The global configuration of the entire ecosystem is a point in a higher-dimension configuration state space, and the entire ecosystem is moving along an evolutionary / innovation trajectory. This problem is important to understanding markets and ecosystems and includes not only issues of development of new system types, but also rates of production and distribution across global supply networks, as a part of the overall innovation model. The diffusion of system types (species, product types, technologies) across the population may be studied in this way. The population perspective has been studied at length in diffusion of technology (Rogers 2003) and proliferation and limits of biological species populations (MacArthur and Wilson 1967).

CONCLUSIONS AND FUTURE STEPS

1. Theories of optimal control and optimal estimation are based in state space and become more applicable to innovation strategy when explicit system models are used to express system configuration.
2. Geometrization of formal spaces, already a source of major insights in the history of STEM, when applied to the innovation domain brings insight and understanding to planning and executing system innovation.
3. Heuristic practices for innovation strategy, agility, risk management, and learning may be enhanced by the use of mathematical system models of life cycle trajectories over innovation cycles.
4. For learning to be effective, the products of learning must be built into the roles that will perform future tasks to be informed by that learning—"lessons learned" filed in reports or searchable databases are not really learned in an effective sense.
5. Use of models does not replace human judgment but enhances it in much the same way that STEM has advanced other human-managed activities, adding science and math-based foundations to previously intuitive practices.
6. Quantitative understanding of agile, fail-fast and recover early, lean, and experiment-based innovation methods is enhanced by viewing these through the lens of trajectory in configuration space. Implications for future pursuit include:
7. How automated engineering tooling can be enabled to assist innovation teams by improving their decision-making around selection of activities;
8. Further exploitation of the historical work of (Pontryagin et al. 1962; Bellman 1957, 1959; and Kalman 1960);
9. Extension of the mathematical theory by moving to populations, applicable to markets and other ecologies;
10. Incorporation of model verification, validation, and uncertainty quantification (VVUQ), and related application of learned system patterns (PBSE);
11. Enhanced visualization of product life cycle trajectories;
12. Simulation of innovation as a dynamical system. ■

ACKNOWLEDGEMENTS

The INCOSE ASELCM discovery project, led by Rick Dove, has provided valued motivation for a stronger theory of systems of innovation in the presence of uncertainty and change.

REFERENCES

- Anderson, Eric T. and Duncan Simester. 2011, "A Step-by-Step Guide to Smart Business Experiments." *Harvard Business Review*, March. https://hbr.org/2011/03/a-step-by-step-guide-to-smart-business-experiments.
- Bellman, R. E. 1957. *Dynamic Programming*. Princeton, US-NJ: Princeton University Press.
- Bellman, R. E, and R. E. Kalaba. 1959. "Dynamic Programming and Feedback Control." RAND Corp.
- Box, George E. P. 2013. *An Accidental Statistician*. New York, US-NY: Wiley.
- Boyer, C. B. 1959. "Descartes and the Geometrization of Algebra." *The American Mathematical. Monthly* 66 (5): 390–393.
- Bryson, A. E., Jr. 1967. "Applications of Optimal Control Theory in Aerospace Engineering" Journal of Spacecraft and Rockets 4 (5): 545-553.

- Bryson, A. and Y. Ho. 1967. "Lecture Notes on Optimization, Estimation, and Control." Harvard University Division of Engineering and Applied Physics, Cambridge, US-MA.
- Bryson, Arthur, and Yu-Chi Ho. 1975. Applied Optimal Control: Optimization, Estimation, and Control, New York-US: Taylor & Francis.
- Christensen, C., C. Dyer, and H. Gregersen. 2011. The Innovators DNA: Mastering the Five Skills of Disruptive Innovators, Boston, US-MA: Harvard Business Review Press.
- Clarke, Ben. 2016. "Why These Tech Companies Keep Running Thousands of Failed Experiments." Fast Company, 26 September.
- Dove, R., and R. LaBarge. 2014. "Fundamentals of Agile Systems Engineering—Part 1" and "Part 2." Papers presented at the 24th Annual International Symposium of INCOSE, Las Vegas, US-NV, 30 June – 3 July.
- Dove, R., et al, 2016, *Proceedings of the INCOSE 2016 Socorro Systems Summit*. http://www.incose.org/ChaptersGroups/Chapters/ChapterSites/enchantment/library-and-resources/socorro-systems-summit—2016-oct-28-29/proceedings.
- Fisher, Ronald. 1971. *The Design of Experiments* Ninth Edition. New York, US-NY: Macmillan.
- Forsberg, Kevin, Hal Mooz, and Howard Cotterman. 2000. Visualizing Project Management: A Model for Business and Technical Success. New York, US-NY: John Wiley and Sons.
- Friedenthal, S., et al. 2015. "A World In Motion: Systems Engineering Vision 2025." San Diego, US-CA: INCOSE.
- INCOSE MBSE Patterns Working Group. 2015. "Pattern-Based Systems Engineering (PBSE), Based On S*MBSE Models." http://www.omgwiki.org/MBSE/doku.php?id=mbse:patterns:patterns_challenge_team_mtg_06.16.15.
- INCOSE Product Line Engineering Working Group. 2015.
- http://www.incose.org/ChaptersGroups/WorkingGroups/analytic/product-lines
- ISO/IEC/IEEE 15288. 2015. Systems Engineering—System Life Cycle Processes. Geneva, CH: International Organization for Standardization.
- Kalman, Rudolf. 1960. "A New Approach to Linear Filtering and Prediction Problems." Transactions of the ASME, Journal of Basic Engineering 82:34–45.
- King, Jeffrey, and Mark Karpenko. 2016, "Estimation of Optimal Control Benefits Using the Agility Envelope Concept." *Advances in the Astronautical Sciences Spaceflight Mechanics* Vol 155.
- Kline, W., A. Bernal, M. Simoni, and W. Schindel. 2017. "Development of Enhanced Value, Feature, and Stakeholder Views for a Model-Based Design Approach." To appear in Proc. of the 2017 *American Society of Engineering Education*, Columbus, US-OH.
- Koch, Richard. 1998. The 80/20 Principle: the Secret of Achieving More with Less. New York, US-NY: Currency Books, Doubleday.
- Kohavi, R., T. Crook, R. Longbotham, B. Frasca, R. Henne, R. Ferres, and T. Melamed. 2009. "Online Experimentation at Microsoft." http://www.exp-platform.com/documents/expthinkweek2009public.pdf.
- Levi, Mark. 2014. Classical Mechanics with Calculus of Variations and Optimal Control. New York, US-NY: American Mathematical Association.
- MacArthur, Robert H., and Edward O. Wilson. 1967. *The Theory of Island Biogeography*. Princeton, US-NJ: Princeton University Press.
- Manzi, James. 2012. *Uncontrolled: The Surprising Payoff of Trial-and-Error for Business, Politics, and Society*. New York, US-NY: Basic Books.
- Martinovich, V. 1988. "Quantifying Aircraft Agility Using Minimum-Time Maneuvers." MS Thesis, Iowa State University (Ames, US-IA).
- Peterson, T., J. Fuchs, and W. Schindel. 2017. "Model-Based Transformation: Planning and Assessment Instrument." INCOSE Corporate Advisory Board Report, INCOSE Annual Workshop, Torrance, US-CA. 28-31 January.
- Pontryagin, L. S., V. G. Boltyanskii, R. V. Gamkrelidze, and E. F. Mishchenko. 1962. *The Mathematical Theory of Optimal Processes*. English transl. New York, US-NY: Interscience.
- Pyster, A. and D. H. Olwell, eds. 2013. The Guide to the Systems Engineering Body of Knowledge (SEBoK) v. 1.1.2. Hoboken, US-NJ: The Trustees of the Stevens Institute of Technology. www.sebokwiki.org/.
- Reinertsen, D. 2009. *Principles of Product Development Flow: Second Generation Lean Product Development*. Redondo Beach, US-CA: Celeritas Publishing.
- Ries, Eric. 2011. *The Lean Startup: How Today's Entrepreneurs Use Continuous Innovation to Create Radically Successful Businesses*. New York, US-NY: Crown Business.
- Rigby, D., J. Sutherland, and H. Takeuchi. 2016. "The Secret History of Agile Innovation." *Harvard Business Review* 20 April.
- Rogers, Everett. 2003. *Diffusion of Innovations* Fifth Edition. New York, US-NY: Free Press.
- Schindel, W. 1972. "Linear Estimation: The Kalman-Bucy Filter." MS Thesis, Rose-Hulman Institute of Technology (Terre Haute, US-IN). http://scholar.rose-hulman.edu/math_grad_theses/1/.
- Schindel, W. 2005. "Requirements Statements Are Transfer Functions: An Insight from Model-Based Systems Engineering." Paper presented at the 15th Annual International Symposium of INCOSE, Rochester, US-NY, 13-16 June.
- Schindel, W. 2015. "System Life Cycle Trajectories: Tracking Innovation Paths Using System DNA." Paper presented at the 25th Annual International Symposium of INCOSE, Seattle, US-WA, 13-16 July.
- Schindel, W. 2016. "Got Phenomena? Science-Based Disciplines for Emerging System Challenges." Paper presented at the 26th Annual International Symposium of INCOSE, Edinburgh, GB-SCT, 18-21 July.
- Schindel, W., and R. Dove. 2016. "Introduction to the Agile Systems Engineering Life Cycle MBSE Pattern." Paper presented at the 26th Annual International Symposium of INCOSE, Edinburgh, GB-SCT, 18-21 July.
- Schindel, W., S. Peffers, J. Hanson, J, Ahmed, and W. Kline. 2011. "All Innovation is Innovation of Systems: An Integrated 3-D Model of Innovation Competencies," in *Proc. of American Society for Engineering Education Annual Conference and Exposition*, Vancouver, BC, 26-29 June.
- Schrage, M. 2014. *The Innovator's Hypothesis: How Cheap Experiments Are Worth More Than Good Ideas*. Cambridge, US-MA: MIT Press.
- Sheard, S., S. Cook, E. Honour, D. Hybertson, J. Krupa, J. McEver, D. McKinney, P. Ondrus, A. Ryan, R. Scheurer, J. Singer, J. Sparber, and B. White. 2016. "A Complexity Primer for
- Systems Engineers." INCOSE Complex Systems Working Group, downloaded March 24,
- 2017, from https://incose.ps.membersuite.com/onlinestorefront/BrowseMerchandise.aspx.
- Simkins, A., R. de Callafon, and W. Todorov. 2008. "Optimal Trade-off Between Exploration and Exploitation." Proc. of the American Control Conference, Seattle US-WA, 11-13 June.

> continued on page 52

What Is the Smallest Model of a System?

William D. Schindel, schindel@ictt.com

[Editor:This paper for systems engineering foundations refers to the *Systems Engineering Handbook* 3rd edition (Copyright 2010 by the International Council on Systems Engineering), ISO 15288:2002, and the Systems Engineering Vision 2020 published by INCOSE in 2007.]

■ **ABSTRACT**

How we represent systems is fundamental to the history of mathematics, science, and engineering. Model-based engineering methods shift the nature of representation of systems from historical prose forms to explicit data structures more directly comparable to those of science and mathematics. However, using models does not guarantee simpler representation—indeed a typical fear voiced about models is that they may be too complex.

Minimality of system representations is of both theoretical and practical interest. The mathematical and scientific interest is that the size of a system's "minimal representation" is one definition of its complexity. The practical engineering interest is that the size and redundancy of engineering specifications challenge the effectiveness of systems engineering processes. INCOSE thought leaders have asked how systems work can be made 10:1 simpler to attract a 10:1 larger global community of practitioners. And so, we ask: What is the smallest model of a system?

INTRODUCTION AND BACKGROUND: SIZE MATTERS!

Representation size, purpose, traditions. This paper discusses possible (and potentially least) upper bounds on the sizes of effective representations of systems, *for the purposes of systems engineering*. Compared to traditional systems engineering approaches, it draws more directly on scientific traditions for representing behavior as physical interaction. Systems engineering is still young, and its connections to supporting sciences is still evolving rapidly.

Language and compression. This subject may appear to be related to the language used to describe systems, and an interesting thread in the mathematical study of description length is whether minimality is in a sense independent of language (Chaitin 2005, Grunwald, Li and Vitany 1997). In any case, systems modeling languages such as SysML® and its predecessors provide valuable assets for the movement to model-based methods (SysML Partners). Our subject here is not the machinery of these specific modeling languages, but the systems ideas that minimal models must address. When used for system families (product lines, ensembles), the representation described here is subject to significant compression by the use of patterns. This turns out to provide powerful insights about approaches to major practical reductions in the size of systems engineering descriptions and processes, and about ongoing future evolution of domain languages over time. These dynamics also suggest that such patterns can be understood as emergent when the interaction rules of the systems engineering process are properly arranged.

Practical representation challenges of traditional systems engineering. Traditional system documentation of concept of operations (CONOPS), system requirements, design specifications, failure mode and effects analysis (FMEA), test plans, operations and maintenance procedures, and other task-specific system representations over the life cycle of a system can exceed thousands of pages. This does not encourage the engagement of a 10:1 larger global community of systems practitioners. Systems engineers may argue that system risks justify these extensive descriptions, but the effectiveness of these representations may be questioned in light of the following typical experiences: A requirements document, read by three systems engineers, produces three interpretations of its meaning — quite a different experience from three electrical engineers interpreting a properly constructed electrical schematic diagram. Whereas the discovery of an ambiguity in a schematic "blueprint" is considered exceptional (or even machine-checkable in some cases) ambiguities in "system" requirements documents are commonplace and frequently tolerated as the state of the systems art. Determining completeness and consistency of (or otherwise interpreting) a specification document is frequently a

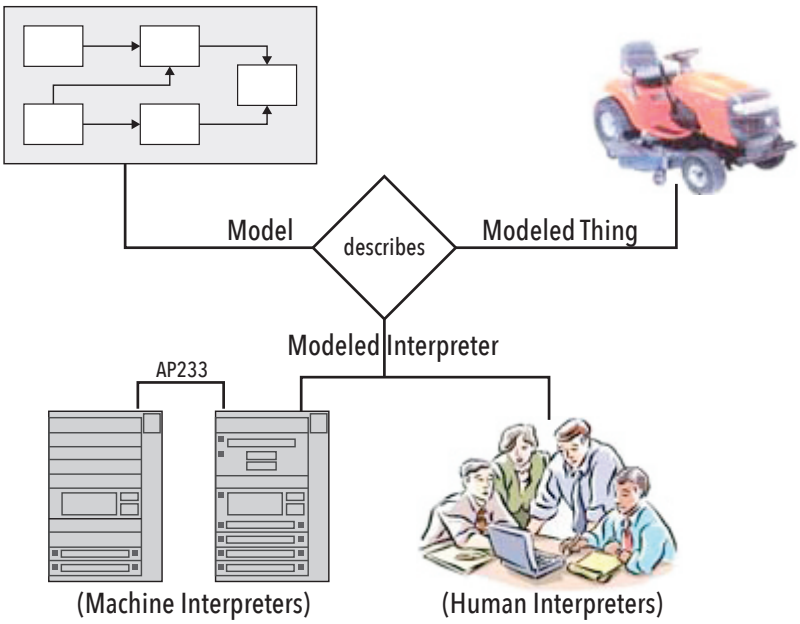


Figure 1. The setting for model-based systems engineering

highly subjective assignment, requiring very experienced human reviewers. Model-based representations are a hoped-for way to address this challenge, but it is not yet obvious whether these are sufficient for an order-of-magnitude positive benefit to the overall systems process.

Systems engineering process versus systems engineering data. This paper's perspective will shift between the systems engineering process (a system in itself), versus information about the target system (which flows through the systems engineering process system), and how the two are related. The systems engineering process is frequently described (ANSI/EIA-632-1998, ISO/IEC 15288 2002, Haskins 2010), but the system representations it produces and consumes (our main subject here) remain a key challenge. We argue that the target systems information is the more fundamental issue to solve, after which the resulting implications for the systems engineereing process can be addressed in a new light.

Complexity science. Complexity, a seemingly intuitive idea, has become the subject of formalization and study, including both the natural and human-engineered world. Initial efforts sought a theoretical basis for expressing complexity measures or otherwise understanding complexity, including the size of minimum system descriptions (Li and Vitany 1997, Chaiten 2005, Kauffman 2000). They have more recently turned to the practical implications of emergent complexity science for engineering processes (Bar-Yam 2003b, 2005, Braha et al. 2006, Kuras and White 2005, Schindel 1996). Some efforts have studied minimal information required to describe a system, as a measure of its complexity. Others have introduced "complex systems engineering" (CSE) terminology in connection with understanding engineering problems or classifying systems, in situations such as highly interconnected systems (networks), adaptive systems, systems embedding humans, issues of scale and scope, ideas about types of emergence, or engineering project failures (Braha et al. 2006, Bar-Yam 2003b). Some studies have focused from the outset on the problems of human engineering or other organic intentional processes in connection with complex engineered systems (Ashby 1957, INCOSE HSIG). There is a growing awareness of connections between systems engineering and systems science. INCOSE formed the System Science Enabling Group, and later the Systems Science Working Group (INCOSE SSWG), in recognition of the connection between systems science and systems engineering.

System patterns. Ideas of "patterns" have a number of connected roots in science and engineering. Pattern recognition and classification have a mathematical theory and engineering practices (Duda 2001). Patterns in engineered systems were recognized in building architecture, later inspiring software engineers, and more recently systems engineers (Alexander 1977, Gamma et al. 1995, Haskins 2005, Cloutier and Verma 2007, Schindel 2005b). Initially expressed using traditional engineering structures (for example, prose templates), patterns were later combined with model-based systems engineering (MBSE) to lead to pattern-based systems engineering (PBSE) (Schindel and Smith 2002, Schindel 2005b).

CONSTRUCTING EFFECTIVE AND EFFICIENT REPRESENTATIONS

Using models. Model-based representations have a traditional engineering role in verifying that designs will satisfy requirements, or otherwise representing system behavior (Karayanakis 1993). More recently, model-based representations have been used to represent system requirements (Mellor 2002, INCOSE MBSE, Schindel 2005a, SysML Partners, Estafan). In the earlier and more established design verification case, "model" frequently refers to mathematical descriptions of system physical make-up, often modeling from first principles to create mathematical descriptions that can be analyzed or simulated. In the more recent system requirements case, "model" extends this idea to describe desired functional behavior.

In both cases, the term "model" means a *formal* (according to agreed upon rules), *explicit* (core content not implicitly depending on other assumed knowledge), and *unambiguous* (not subject to multiple interpretations) description. As shown in Figure 1, there are three components in a model-based engineering setting: The model, the system modeled, and the model interpreter(s). We want the model to be interpreted with desired process outcomes (for example, easy, consistent, and unambiguous interpretation, optimality of design, etc.). Global efforts (ISO 10303 AP233 and Mellor 2002) are working toward the exchange and interpretation of model data by machines and people, for purposes of simulation, procurement, fabrication, code generation, etc.

The "third role" (model interpreter) in Figure 1 has vital significance here. The effectiveness of a model means how well it serves the purposes of the model interpreter. If we expect to engage a 10:1 larger community of systems practitioners, and make the systems process 10:1 easier, then we must learn how to make the model interpreter's tasks easier and more appealing, and for a much larger global population. If we only develop automated approaches to deluge the human model interpreter with information, we won't have the outcome needed.

A metamodel. A metamodel is a model of other models — a framework or plan governing the models that it describes. We utilize the S*Metamodel (summarized by Figure 2), a relational/object information model used in the Systematica™ methodology to describe requirements, designs, and other (verification, failure analysis, etc.) information in S*Models. These may be represented in SysML®, database tables, or other languages. We have applied these to systems engineering in mil/aero, transpor-

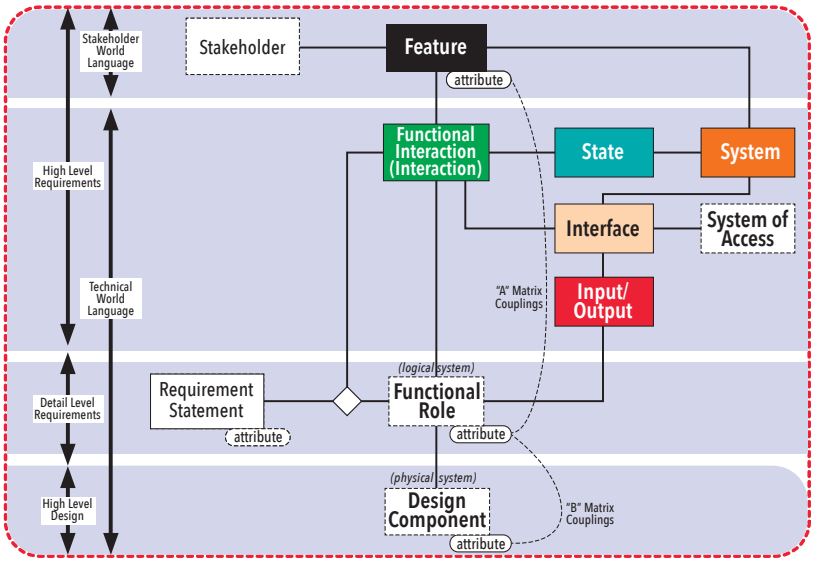


*Figure 2. A summary view of the S*Metamodel*

tation, communication, medical and health care, consumer products, construction, manufacturing, and as a framework for educating new engineers (Gunyon et al. 2010; Bradley et al. 2010; Schindel and Smith 2002; Schindel 2002, 2005b; and Ahmed et al. 2011).

S*Models describe the external (black box) behavior of target systems twice — once in the subjective (stakeholder) language of stakeholder-valued behavior, and again as more objectively-described technical behaviors.

Stakeholders features. S*Models represent system stakeholder features as explicit objects. For example, some of the features of an oil filter are represented in Figure 3:

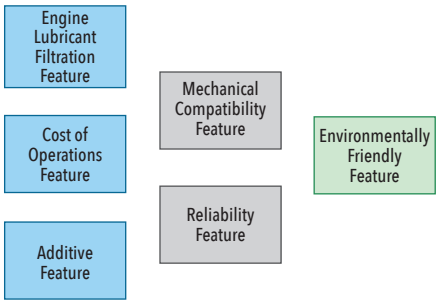


Figure 3. Features of an oil filter

We could simply claim that a minimal model of an engineered system must include a (feature) model of all the things valued by all the system's stakeholders. However, there is more to this than meets the eye. Features have a way of creeping into many different engineering conversations and artifacts, not always recognized for their redundancy. Note that every design decision, every trade-off, every value engineering or project argument should ultimately depend upon no less than, *but also no more than*, the feature model. (This is based upon the practice of including all significant stakeholders and their features in the feature model.) If we find a compelling argument for why technology X or architecture Y is the right (or wrong) choice, the reason why can only be to better accommodate the stakeholder features — trade space is exclusively "scored" in the metrics of these features. A common mistake is to defend choices in technical trade-off spaces that are short of the actual stakeholder feature space.

Suppose further that we are performing an FMEA. It turns out that the only "effects" (the E part) that can appear in an FMEA are *failures to deliver on the promise of a stakeholder feature*. As soon as we know the feature space of system, before a design has been synthesized, we can already fill out the "effects" column of the FMEA analysis (Schindel 2010). However, it is not universal practice to align or audit FMEA and stakeholder feature models.

Feature space is integrated with technical requirements space by the negotiation of the features-interactions relationships — to begin with, a two-column table negotiated jointly by representatives of the stakeholders and the technical community. Feature space is typically of lower dimension than the more technical spaces of system requirements or design. This means that once we have constructed an integrated feature-interactions-roles-requirements model (traced by Figure 2), we can "configure" (automatically populate) good starting point draft requirements configurations: populating lower dimension features can "automatically" populate higher dimension requirements through the constraints of the model.

We have repeatedly seen the use of feature models dramatically improve alignment and facilitate constructive discussion of cross-functional teams. For example, a powerful use of feature space is to express impact assessments on the introduction of new technologies into operations environments, or to express system long range or facility master plans first in terms of features (stakeholder capabilities) planned and only second in terms of the equipment, technologies, or projects that will implement them. Likewise, risk to stakeholders (whether financial risk, schedule risk, technical risk, or otherwise) is represented by features.

All this suggests that feature models are often under-utilized in the rush to technical requirements. Note that feature models are formal even though they are in the (subjective) language of stakeholders. There is a difference between informally stated stakeholder "needs" (in the original voice of the customer) and formally translated (but still stakeholder language and concept) features. A quick pass through "stakeholder needs" on the way to technical requirements is less than the minimal Features model we are suggesting here.

Feature space is an interesting place. It is the gateway to other communities beyond our engineering organizations, and for that reason may be seen as a strange or unfamiliar language and environment. But it represents improved connection to those who pay the bills, buy the products, or whose lives depend on the engineered system. Bridging this cultural gap may be challenging but is the reason that S*Models are dual rooted in both of the "two cultures" (C.P. Snow, S. J. Gould). When INCOSE thought leaders advocate that we look for ways to engage order-of-magnitude larger segments of the global community in systems work, modeled feature space, in model views appropriate to the viewers, are a related enabler.

Interactions. S*Models represent physical interactions as explicit objects at the very core of systems engineering. For example, Figure 4 shows interaction objects for an oil filter—these summarize the physical interactions of an oil filter with its environment, over its life cycle.

Interaction models exist at two levels of detail. The high-level interaction model simply consists of the name and definition of the interaction, a list of the parties that participate (play roles) in the interaction, and the major attributes of the interaction. These named interactions also appear within the system's state model, and that combination very compactly expresses the overall modeling of the system's behavior with its environment, over its life cycle. The detail level interaction model includes

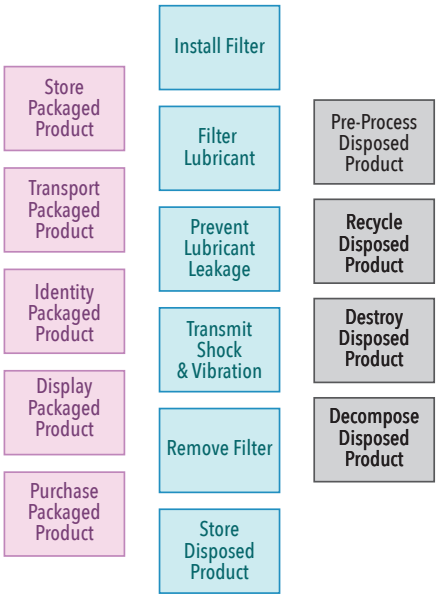


Figure 4. Interactions of an oil filter

an interaction diagram (of which there are many specific forms in SysML or other modeling languages) for each interaction, showing the input-outputs exchanged between the interacting actors, and including the requirements statements that describe the roles in the form of "nonlinear transfer function" relationships between the inputs and outputs (Schindel 2005). Refer to Figure 5.

or more components playing logical roles. The "emergent" properties of the interaction are associated with the whole, not any single component. Behaviors of individual components are described by requirements statements as input-output characteristics of their roles (Schindel 2005a). Notice the difference in perspective of Figure 6.

By now it is well-known that simple behaviors by individual components (or "agents", in the popular parlance), when they interact with each other, may lead to "emergence" of surprisingly more complex behavior by the combined system (for example, the three body problem, cellular automata, swarms, traffic, etc.). The difference between the simple "rules" (behavior of the actors) and the more complex emergent system behavior is nothing more and nothing less than the difference between describing a functional role in an interaction and the interaction as a whole—it is a difference of night and day.

We have repeatedly observed a profound practical difference between systems engineering modeling interactions as illustrated by Figures 5 and 6, versus modeling of SIPOC required behavior as in Figure 6. We have seen this difference have major practical impact in numerous systems engineering projects, in which the modeler either did or did not model the whole interaction, including "what the operator did" (Schindel 2006), "what the material did" (Schindel 2011), or other actor behaviors.

Minimality of representation. The S*Metamodel arose over time from the research question, "What is the smallest amount of information required to describe system level requirements and design?", combined with practice application. We won't reproduce here the formal argument for minimality of S*Models—interested readers may contact the author. However, a summary of that argument is:

- The sufficiency of S*Models of requirements and designs is argued, with respect to intended use of the information. Here the *uses of systems engineering information* enter, including considerations of risk and opportunity.
- The minimality of S*Models is established by showing that no metaclass (see Figure 2) of information in an S*Model is redundant with information in another metaclass, and showing that omission of any component results in loss of sufficiency—including classes versus instances.

This argument makes use of a mapping of which S*Model components (grouped across the top of Table 1) are needed for the different SE process areas (summarize by the Table 1 rows).

This table can be constructed for the various systems enginering process areas of ISO15288 or the INCOSE *Systems*

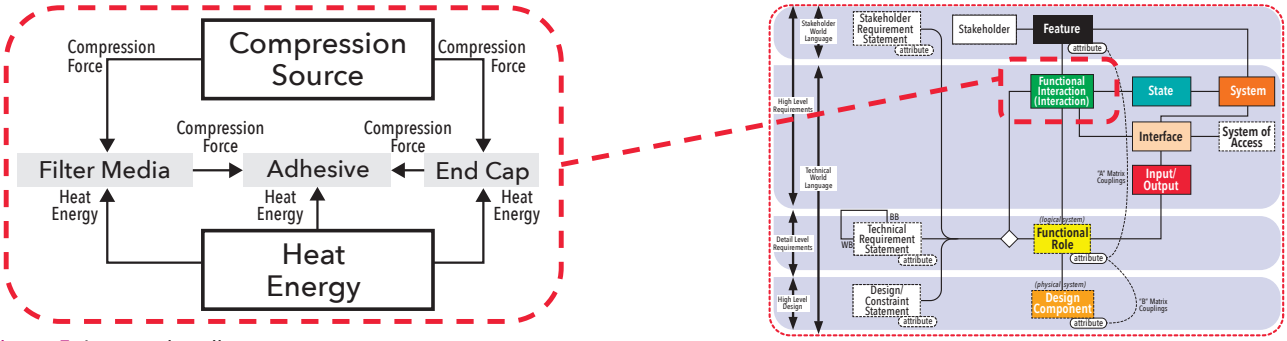


Figure 5. Interaction diagram

Interaction models go to the heart of what we mean by "system" in the engineering and scientific world and expresses ideas of emergence. By "system," we mean a collection of interacting components. By "interact," we mean that one component impacts the state of another component. By "state," we mean a property of a component that impacts its current or future behavior. By "behavior", we mean a component's interactions with other components. This is the intentionally circular, relational perspective of the trained scientist, engineer, or mathematician that has helped describe the natural world since Newton. In this perspective, an interaction is holistic, with two

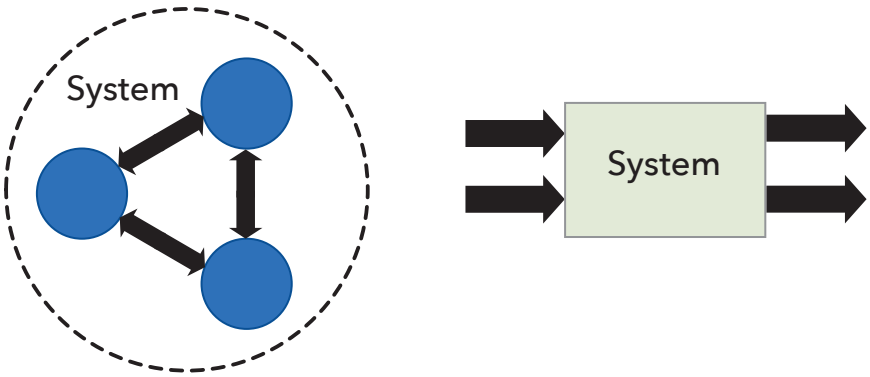


Figure 6. Two different starting points: systems as interacting components versus a SIPOC perspective

Table 1. Systems engineering process areas vs. metamodel information areas

Systems Engineering Area	Grp1	Grp 2	Grp 3	Grp 4	Grp 5
HLR	X				
DLR/BB	X	X			
DLR/WB	X	X			
HLD	X	X	X		
FMEA	X	X	X	X	
TST	X	X	X	X	X

Engineering Handbook. However, later in this paper we will also discuss an alternate way to view systems engineering process areas. (Why would we want to do that? The answer depends on whether we expect a much larger global population to become traditional systems engineers and take up the traditional systems engineering processes.)

The above minimality argument is "constructive": Rather than arguing that a minimal model exists, we actually construct it—not the case in most algorithmic information theory. However, this argument does not assert uniqueness: There may be other models no larger that also represent the same system.

MODEL VIEW; USEFUL REDUNDANCY.

A familiar challenge is that different "systems engineering documents" may be inconsistent with (contradict) each other: This is because they contain redundant information. As documents evolve, that consistency must be maintained to be consistent across the documents. Refer to Figure 7. This issue also occurs within single documents (self-consistency). There are good (task-oriented) reasons why these documents should be redundant—but not why they should be inconsistent.

This is one reason why database tools are powerful in systems engineering. Properly used, they can generate different "views" (documents, etc.) from the common underlying data model, thereby improving their consistency (see Figure 8).

The S*Model goes farther, by pointing out redundancies not always recognized; for example,

- FMEA functional failures vs. requirements (counter-requirements) (Schindel 2010)
- FMEA failure effects vs. stakeholder features (noted earlier above)
- ICDs vs. system requirements
- CONOPS and use cases vs. system requirements, features.

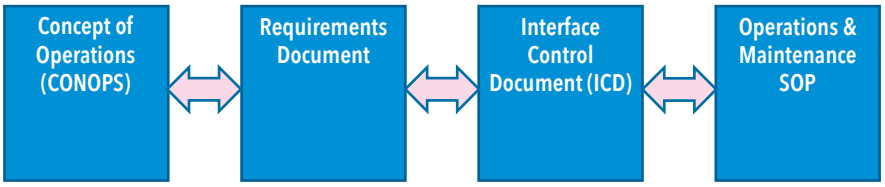


Figure 7. Redundant documents—consistent or inconsistent?

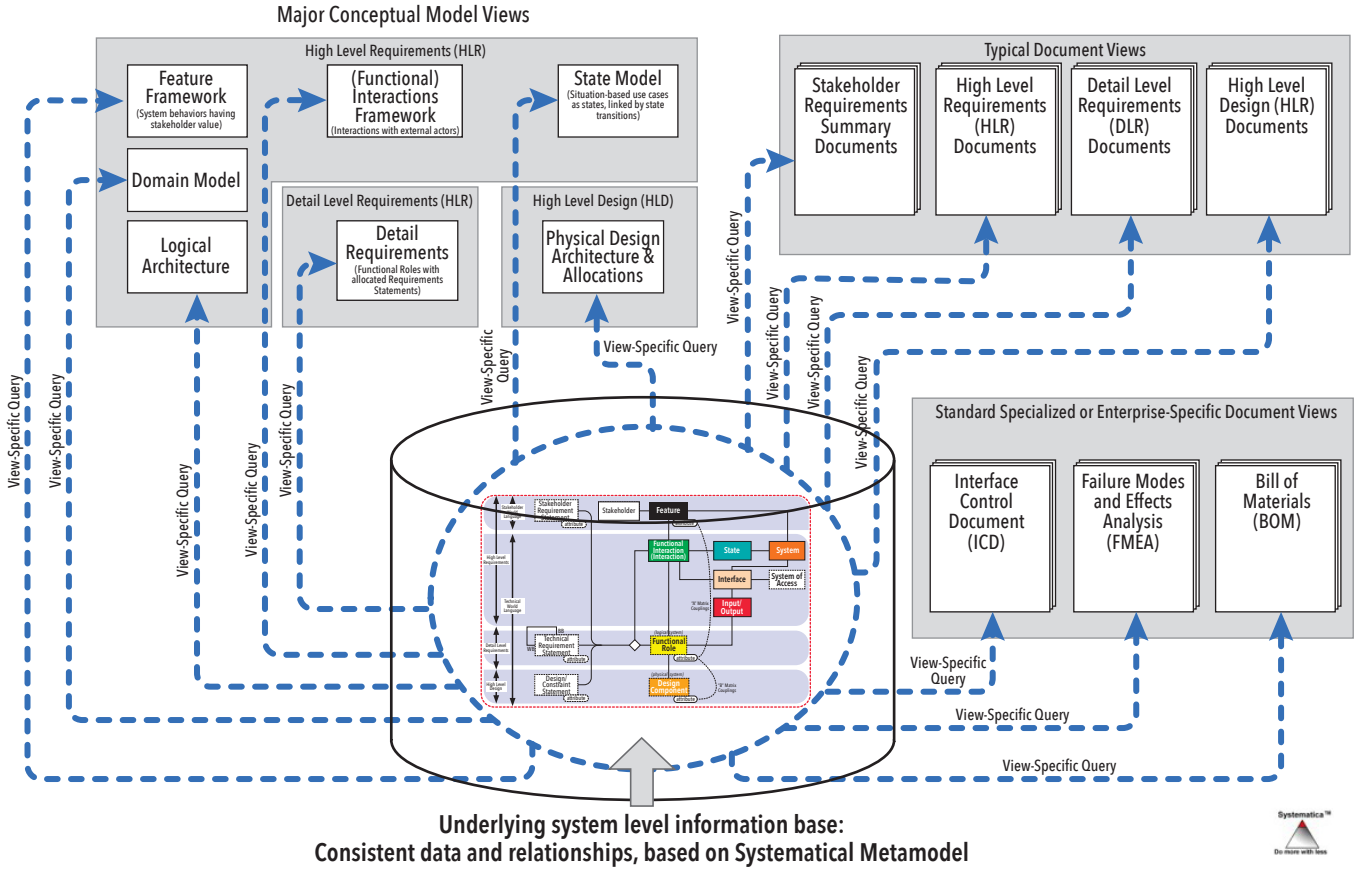


Figure 8. Generation of (redundant) views from a non-redundant database

Such "redundancies" are really deep insights that make model construction easier and reinforcing: We can still produce all these views, but with less effort and greater consistency.

Measures of model complexity. Models communicate information, as quantified in communication theory (Shannon 1963). More recently, complexity of objects has been quantified in algorithmic information theory (AIT or Kolmogorov complexity) using the "smallest program" capable of constructing the object or its behavior (Li and Vitany 1997, Chaiten 2005). The minimality of S*Models (measured in bits) also have several practical sides:

- Clarifying "too small" versus "big enough" models: The S*Metamodel reminds us of types of systems engineering information that, if omitted, will leave us with an incomplete description of a subject system's requirements, design, or connecting relationships. A practical example is the use of states in a requirements model, reminding us that for any requirement statement, "when does this requirement apply?" is a fair (and often not explicitly answered) question. We may omit this information for pragmatic reasons, but are reminded of what we have not communicated.
- Reducing redundancy and associated inconsistency: Although documents or other task-oriented views generated from an S*Model may be redundant, the information in an S*Model is not. The consistency of a large number of redundant derived documents and views is easier to maintain or check against a single minimal model.

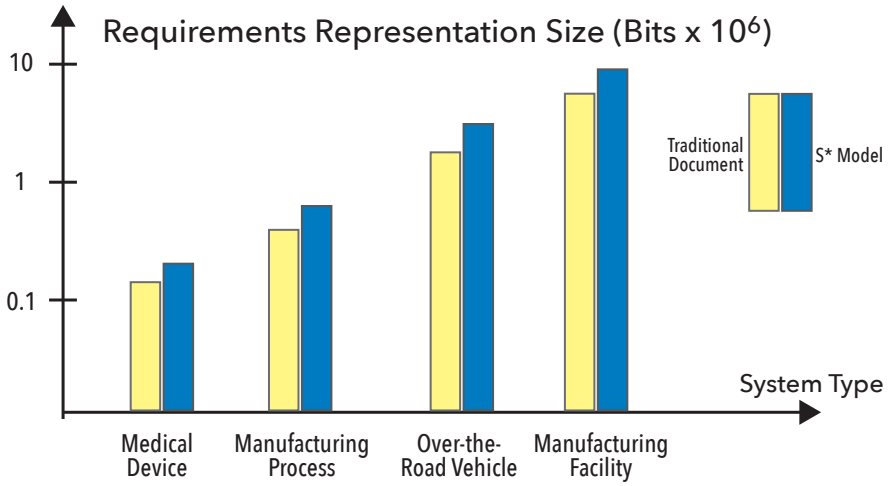


Figure 9. Typical sizes for models and traditional systems engineering documents

So, how big? How does an S*Model-based compare in size to a traditional systems engineering prose-based description? A practical discovery is that a typical S*Model of technical requirements is more complete than a corresponding traditional technical requirements document. Being more complete, it is bigger, not smaller! Figure 9 illustrates some typical sizes. Keep in mind the original question was: What information is essential?

USING PATTERNS TO COMPRESS MODELS

The "starting from scratch" systems engineering process delusion. One of the most significant causes of perceived complexity of the systems engineering process is the fact that most descriptions of the process seem to (implicitly) involve an assumption (judging from the steps they describe) that is nearly always false for real projects — that the project is "starting from scratch" in a "clean sheet" engineering project on a system for which there are no significant historical precedents. Accordingly, the process systematically seeks out the needed information and processes it into a form usable by the project (ISO/IEC 15288, INCOSE *Systems Engineering Handbook*).

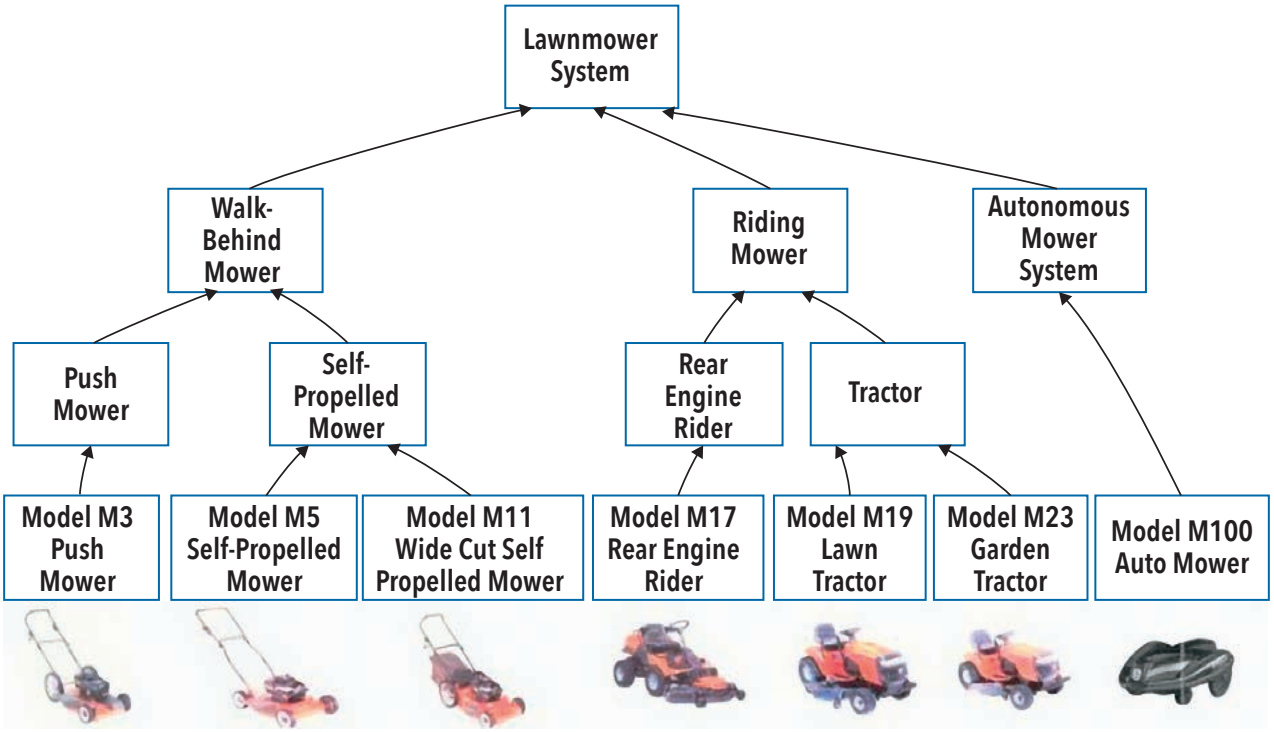


Figure 10. Families of systems—whether generations or product lines

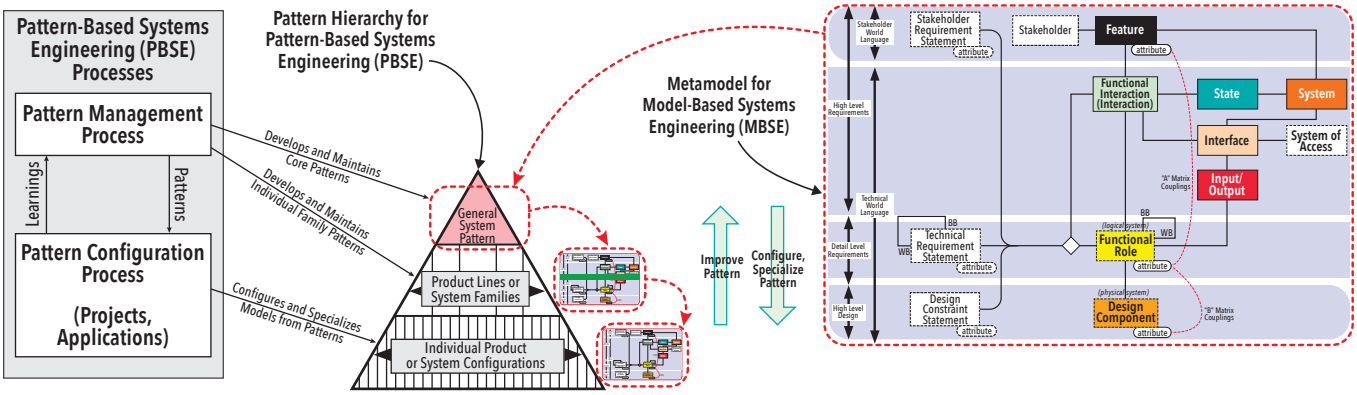


*Figure 11. An S*Pattern is a configurable, re-usable S*Model*

On the contrary, real projects are most often concerned with engineering similar (but different) systems across different product generations, applications, configurations, or market segments. At the very least, we are typically engineering whiz-bang product X as the latest improvement in a long line of previous products in the same domain — but with some new differences, big or small. In some cases, we are even planning a product line of related products as a whole (see Figure 10).

In spite of this reality, very little of the descriptions of the systems engineering process is typically about *more efficiently leveraging what we already know* about the target systems. Typically, these descriptions make some mention of consulting documents or lessons learned about similar projects, but very rarely is there a procedural discipline *focused specifically on engineering of what we could call "variable sameness*." Something more than a database of useful past requirements or cloning the last project document from the engineer's desk drawer (a dominant paradigm) is suggested here — an equivalent to perturbation theory in mathematics.

Pattern-based systems engineering (PBSE). Over several decades, we have developed and practiced what we call pattern-based systems engineering (PBSE) across a range of domains, including carrier grade telecommunications, engines and power systems, automotive and off road heavy equipment, telecommunications, military and aerospace, medical devices, pharmaceutical manufacturing, consumer products, and advanced manufacturing systems (Schindel and Smith 2002, Schindel 2005b, Bradley et al. 2010). Engineers in all of these and many other domains spend most of their company's engineering resources developing or supporting systems that virtually always include major content from repeating system paradigms at the heart of their business (for example, core ideas about airplanes, engines, switching systems, etc.). In spite of this, the main paradigm apparent in most enterprises to leverage "what we know" is to build and maintain a staff of experienced technologists, designers, application engineers, or other human repositories of knowledge. There is typically little evidence of a "Maxwell's Equations" of first principle-based discipline of "variable sameness" in the engineering of these systems.

Although engineering "patterns" already have precedent in systems and software engineering (Gamma 1995, Alexander 1977, Haskins, 2005, Cloutier and Verma 2007), these are often relatively informal approaches to capturing and re-applying certain general ideas, supporting by templates of one sort or another. By contrast, in PBSE what we are doing is to *extend MBSE* through the use of *formally configurable and re-usable systems engineering models*. Specifically, an S*Pattern is a re-usable, configurable S*Model of a family (product line, set, ensemble) of systems.

Pattern configurations. Such patterns are ready to be configured to serve as models of individual systems in projects. "Configured" here is specifically limited to mean that pattern model components are populated/de-populated, and that pattern model attribute (parameter) values are set—both based on configuration rules that are part of the Pattern. Patterns are based on the same metamodel as "ordinary" models.

Because of this disciplined approach to "configuration" as a limited case of specialization, relatively dramatic simplifications can frequently occur in the typical engineering process. A table of configurations illustrates how patterns facilitate compression. The rows of the table represent aspects of the model such as stakeholder features and their attributes, functional roles, requirements attributes, design components, interfaces, etc. (See Table 2).

A different way to organize systems engineering processes: PBSE offers us a different (and potentially simpler) way to view the organization of the systems engineering process areas. Instead of dividing, them by their ISO 15288 type functionality first, we can divide them into two major processes (see Figure 11):

- Pattern management process: Generates the underlying family model, and periodically updates it based on application project discovery and learning.
- Pattern configuration process: Configures the pattern into a specific model for application in a project.

The second of these two processes may well contain what could be viewed as outcome equivalents to the ISO 15288 process areas, but they can be viewed in a much different light if they are first each asking how to produce their products from what is already known (the patterns that govern the target system – not the engineering process). Much of the more complex formal machinery of systems engineering can then be "hidden" in the other process — the pattern management process, in which a much smaller number of people's efforts are leveraged by a larger population in the second process. In this approach, patterns become valued IP, and are sometimes even financially capitalized as a form of "software."

As a start toward "thermodynamics of patterns", the Gestalt Rules (Schindel 1997) describe what it means for a holistic system model to either conform to or not conform to a more general holistic system model. For example, if we develop state models of aircraft over mission profiles that include preparation, take-off, climb, cruise, combat, return, landing, etc., then how can we compare fixed-wing, helicopter, VTOL, civil, and other aircraft?

Compression of models, using patterns. Each column in the table is a compressed system representation with respect to ("modulo") the pattern. The compression is typically very large. The compression ratio tells us how much of the pattern is

Table 2. Pattern configuration table

Lawnmower Product Line: Configuration Table									
		Units	Walk-Behind	Walk-Behind	Walk-Behind	Riding	Riding	Riding Mower	Autonomous
			Push Mower	Mower	Self-Propelled	Rider	Tractor	Tractor	Autonomous
			Push Mower	Self-Propelled	Wide Cut	Rider	Lawn	Garden	Auto Mower
	Model Number		M3	M5	M11	M17	M19	M23	M100
	Market Segment		Small REsident	Medium Resident	Medium Resident	Large Resident	Large Resident	Home Gardn	High End Suburban
Power	Engine Manufacturer		B&S	B&S	Tecumseh	Tecumseh	Kohler	Kohler	Elektroset
	Horsepower	HP	5	6.5	13	16	18.5	22	0.5
Production	Cutting Width	Inches	17	19	36	36	42	48	16
	Maximum Mowing Speed	MPH	3	3	4	8	10	12	2.5
	Maximum Mowing Productivity	Acres/Hr			1.6				
	Turning Radius	Inches	0	0	0	0	126	165	0
	Fuel Tank Capacity	Hours	1.5	1.7	2.5	2.8	3.2	3.5	2
	Towing Feature						x	x	
	Electric Starter Feature				x	x	x	x	
	Basic Mowing Feature Group		x	x	x	x	x	x	x
Mower	No. of Anti-Scalping Rollers		0	0	1	2	4	6	0
	Cutting Height Minimum	Inches	1	1.5	1.5	1.5	1	1.5	1.2
	Cutting Height Maximum	Inches	4	5	5	6	8	10	3.8
	Operator Riding Feature					x	x	x	
	Grass Bagging Feature		Optional	Optional	Optional	Optional	Optional	Optional	
	Mulching Feature		Standard	Factory Installed	Dealer Installed				
	Aerator Feature					Optional	Optional	Optional	
	Autonomous Mowing Feature								x
	Dethatching Feature					Optional	Optional	Optional	
Physical	Wheel Base	Inches	18	20	22	40	48	52	16
	Overall Length	Inches	18	20	23	58	56	68	28.3
	Overall Height	Inches	40	42	42	30	32	36	10.3
	Width	Inches	18	20	22	40	48	52	23.6
	Weight	Pounds	120	160	300	680	705	1020	15.6
	Self-Propelled Mowing Feature			x	x	x	x	x	x
	Automatic Transmiss. Feature							x	
Financials	Retail Price	Dollars	360	460	1800	3300	6100	9990	1799
	Manufacturer Cost	Dollars	120	140	550	950	1800	3500	310
Maintenance	Warranty	Months	12	12	18	24	24	24	12
	Product Service Life	Hours	500	500	600	1100	1350	1500	300
	Time Between Service	Hours	100	100	150	200	200	250	100
Safety	Spark Arrest Feature		x	x	x	x	x	x	

variable and how much fixed, across the family of potential configurations. Refer to Figure 12.

Connection to minimum description length (MDL) theory. In MDL and Kolmogorov complexity theories applied to complexity, there is an idea of the representation of a system "modulo" a certain language used to describe it. Likewise, in PBSE, the configuration of a pattern is a "description" of that system within the space of systems governed by that pattern. If we assume that the pattern itself is already known or accepted, then the configuration information becomes a (much shorter) description of "where in the pattern space the particular configuration is," tying down the degrees of freedom offered by the pattern.

If the language that emerges from a pattern is extremely flexible (for example, English prose), then the degrees of freedom are very large indeed, and the configuration data itself must be extensive. But, if the pattern is based on a construct like the S*Metamodel, then the domain-specific systems engineering language that emerges from that pattern is orders of magnitude more restrictive, and the configuration information is accordingly much simpler and easier to understand, analyze, and communicate.

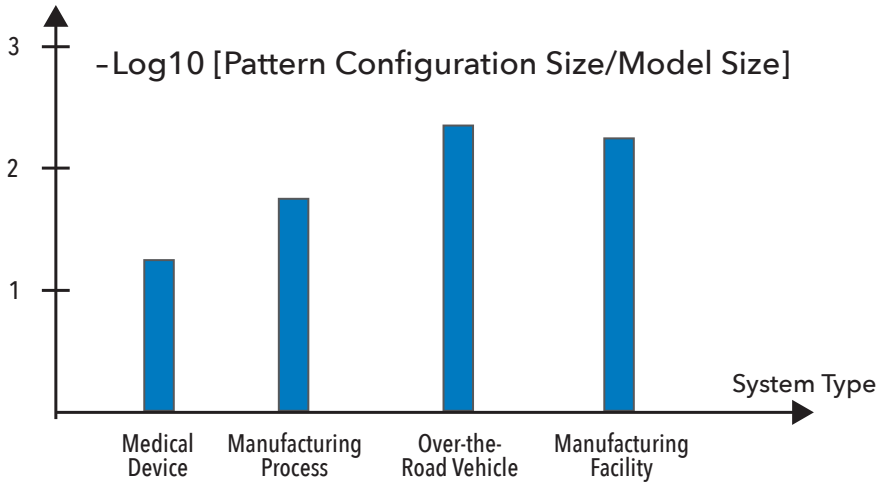


Figure 12. Pattern compression

ALL MODELS ARE CONFIGURATIONS OF MORE ABSTRACT PATTERNS

We arrive at a core idea for simplifying the systems engineering process. Instead of asking how to adopt all the sophisticated machinery of formal PBSE, we can alternatively realize that all models are configurations of more abstract patterns, whether we formalize those patterns are not. Moreover, even non-MBSE engineering projects are in fact creating informal "configurations" of informal "patterns" every day and have been all along. As evidence of this, consider all the "important known stuff" that we don't always write down in projects — the content of industry and enterprise standards comes first to mind. We "invoke" these by reference, but we rarely import explicitly all of their content into our specifications. They become stacks of additional "side" documents that vex designers, suppliers, and others who must conform to them or verify conformance.

What is missing in (most but not all of) these traditional approaches is a sufficient machinery to truly configure these patterns of "external" data for a given project. At best, we might typically see citations of particular sections of these documents that are chosen to apply. More typically, we are left to wonder which parts of these stacks may apply and which do not. By adopting some of the simplest elements of PBSE discipline, once onerous processes can become assets, as we move more rapidly with configuration data, supported by less frequently consulted (but nevertheless available when needed) "pattern" information in these other references.

CONCLUSIONS

1. The specific MBSE and PBSE methods discussed here have been successfully applied across a wide range of domains: transportation, mil/aero, communications, medicine/healthcare, advanced manufacturing, consumer products.
2. The minimum base of information required to perform specific systems engineering process areas is greatly clarified by MBSE metamodel understanding.
3. Minimal MBSE models contain information missing from many projects, causing practical project problems.
4. Minimal underlying models generate the redundancies needed across different task-based artifacts, with greater consistency or less effort to maintain that consistency.
5. Formalization of patterns as configurable models leads to further size compression: configurations.
6. All models are actually configurations of more abstract patterns. Realizing and exploiting this can turn the previous "deadweight" of standards and other external references into powerful assets for accelerating work. ■

REFERENCES

- Ahmed, J., J. Hansen, W. Kline, S. Peffers, and W. Schindel. 2011. "All Innovation is Innovation of Systems: An Integrated 3-D Model of Innovation Competency." To appear in *Proceedings of the 2011 American Society for Engineering Education Annual Conference*, Vancouver, CA-BC, 26-29 June.
- Alexander, Christopher, Sara Ishikawa, Ingrid Fiksdahl-King, and Shlomo Angel. 1977. *A Pattern Language: Towns, Buildings, Construction*. New York, US-NY: Oxford University Press.
- Ashby, W. Ross. 1957. *An Introduction to Cybernetics*. London, GB: Chapman & Hall.
- Bar-Yam, Y. 2003b. "When Systems Engineering Fails—Toward Complex Systems Engineering." *Proceedings of the International Conference on Systems, Man & Cybernetics*, 2: 2021-2028. Piscataway, US-NJ: IEEE Press.
- Bar-Yam, Y. 2005. "About Engineering Complex Systems: Multiscale Analysis and Evolutionary Engineering." *Engineering of Self-Organizing Systems* 2004, LNCS 3464: 16-31, Springer-Verlag.
- Bradley, J, M. Hughes, and W. Schindel, 2010. "Optimizing Delivery of Global Pharmaceutical Packaging Solutions, Using Systems Engineering Patterns." Paper presented at the 20th Annual International Symposium of INCOSE, Chicago, US:IL, 11-15 July.
- Braha, D., A. Minai, Yaneer Bar-Yam, eds. 2006. *Complex Engineered Systems: Science Meets Technology*. Berlin Heidelberg, DE: Springer.
- Chaitin, Gregory. 2005. *Metamath: The Quest for Omega*, New York, US-NY: Pantheon.
- Cloutier, Robert J., Dinesh Verma. 2007. "Applying the Concepts of Patterns to Systems Architecture." *Systems Engineering* (Wiley) 10 (2): 138-154.
- Duda, Richard. O., Peter E. Hart, David G. Stork. 2001. *Pattern Classification* (2nd ed.). New York, US-NY: Wiley.
- Estafan, J. 2008. "Survey of Model-Based Systems Engineering (MBSE) Methodologies." INCOSE MBSE Initiative.
- Gamma, E., R. Helm, Ralph Johnson, J. Vlissides. 1995. *Design Patterns: Elements of Reusable Object-Oriented Software*. Reading, US-MA: Addison-Wesley.
- Gould, S. J. 2003. *The Hedgehog, the Fox, and the Magister's Pox: Mending the Gap between Science and the Humanities*. New York, US-NY: Three Rivers Press.
- Grunwald, P. 2007. *The Minimum Description Length Principle*. Cambridge, US-MA: MIT Press.
- Gunyon, R., and W. Schindel. 2010. "Engineering Global Pharmaceutical Manufacturing Systems in the New Environment." Paper presented at the 20th Annual International Symposium of INCOSE, Chicago, US:IL, 11-15 July.
- Haskins, Cecilia. 2005. "Application of Patterns and Pattern Languages to Systems Engineering." Paper presented at the 15th Annual International Symposium of INCOSE, Rochester, US-NY, 13-16 June.

- Haskins, Cecilia, ed. 2010. *Systems Engineering Handbook Version 3.2*. Seattle, WA: International Council on Systems Engineering.
- INCOSE HSIG web site. https://www.incose.org/communities/working-groups-initiatives/human-systems-integration.
- INCOSE MBSE web site: https://www.incose.org/communities/working-groups-initiatives/mbse-initiative.
- INCOSE SSWG web site: https://www.incose.org/communities/working-groups-initiatives/systems-science
- ISO 10303 AP233 web site. https://segoldmine.ppi-int.com/node/44952.
- ISO/IEC 15288. 2002. Systems Engineering – System Life Cycle Processes. Geneva, CH: International Organization for Standardization.
- Karayanakis, N. 1993. *Computer-Assisted Simulation of Dynamic Systems with Block Diagram Languages*. Boca Raton, US-FL: CRC Press.
- Kauffman, Stuart. 2000. *Investigations*. New York, US-NY: Oxford University Press.
- Kuras, M. L., B. E. White. 2005. "Engineering Enterprises Using Complex-System Engineering." Paper presented at the 15th Annual International Symposium of INCOSE, Rochester, US-NY, 13-16 June.
- Li, Ming, Paul Vitany. 1997. *An Introduction to Kolmogorov Complexity and its Applications* Second edition. New York, US-NY: Springer.
- Mellor, Stephen, Marc J. Balcer. 2002. *Executable UML: A Foundation for Model-Driven Architecture*. Boston, US-MA: Addison-Wesley.
- Schindel, W. 1996. "Systems Engineering: An Overview of Complexity's Impact." Tech Paper 962177, SAE International.
- Schindel, W. 1997. "The Tower of Babel: Language and Meaning in System Engineering." Technical Report No. 973217 SAE International.
- Schindel, W. 2005a. "Requirements Statements are Transfer Functions: An Insight from Model-Based Systems Engineering." Paper presented at the 15th Annual International Symposium of INCOSE, Rochester, US-NY, 13-16 June.
- Schindel, W. 2005b. "Pattern-Based Systems Engineering: An Extension of Model-Based Systems Engineering." TIES tutorial presented at 15th Annual International Symposium of INCOSE, Rochester, US-NY, 13-16 June.
- Schindel, W. 2006. "Feelings and Physics: Emotional, Psychological, and Other Soft Human Requirements, by Model-Based Systems Engineering." Paper presented at the 16th Annual International Symposium of INCOSE, Orlando, US-FL, 9-13 July.
- Schindel, W. 2010. "Failure Analysis: Insights from Model-Based Systems Engineering." Paper presented at the 20th Annual International Symposium of INCOSE, Chicago, US-IL, 11-15 July.
- Schindel, W. 2011. "Systems Engineering for Advanced Manufacturing: Unit Op Insights from Model-Based Methods." Paper presented at the 21st Annual International Symposium of INCOSE, Denver, US-CO, 20-23 June.
- Schindel, William D., Vern R. Smith. 2002. "Results of Applying a Families-of-Systems Approach to Systems Engineering of Product Line Families." Technical Report 2002-01-3086. SAE International.
- Shannon, Claude. 1963. *A Mathematical Theory of Communication*. Champaign, US-IL: University of Illinois Press.
- Snow, C. P. 1960. *The Two Cultures*. Cambridge, GB: Cambridge University Press. pp. 181. ISBN 978-0521457309 (second edition; 1993 reissue).
- SysML Partners web site. http://www.sysml.org/.

ABOUT THE AUTHOR

[Editor: Author biography was current when the paper was initially published in 2011.]

William D. Schindel is president of ICTT System Sciences, a systems engineering company, and developer of the Systematica™ methodology for model and pattern-based systems engineering. His 40-year engineering career began in mil/aero systems with IBM Federal Systems, Owego, NY, included service as a faculty member of Rose-Hulman Institute of Technology, and founding of three commercial systems-based enterprises. He has consulted on improvement of engineering processes within automotive, medical/health care, manufacturing, telecommunications, aerospace, and consumer products businesses. Schindel earned the BS and MS in mathematics, and was awarded the Hon. D.Eng by Rose-Hulman Institute of Technology for his systems engineering work.

Schindel – Innovation, Risk, Agility continued from page 42

- Simmons, G. 2003. *Introduction to Topology and Modern Analysis*. Krieger (Reprint Edition US).
- Simoni, M., E. Andrijcic, W. Kline, and A. Bernal. 2016. "Helping Undergraduate Students of any Engineering Discipline Develop a Systems Perspective." Paper presented at the 26th Annual International Symposium of INCOSE, Edinburgh, GB-SCT, 18-21 July.
- Smaling, Rudolph. 2005. "System Architecture Analysis and Selection Under Uncertainty." Doctoral dissertation, MIT, (Cambridge, US-MA), from http://hdl.handle.net/1721.1/28943.
- Teller, A. 2016. Ted Talk by Astro Teller, 14 April. Alphabet X, retrieved from: www.ted.com/talks/astro_teller_the_unexpected_benefit_of_celebrating_failure.
- Thomke, S. 2003. *Experimentation Matters: Unlocking the Potential for New Technologies for Innovation*. Boston, US-MA: Harvard Business Review Press.
- Walden, D. et al., eds. 2015. *Systems Engineering Handbook: A Guide for System Life Cycle Processes and Activities* Fourth Edition, INCOSE, San Diego, US-CA: Wiley.
- Wiener, N. 1949. "The Extrapolation, Interpolation and Smoothing of Stationary Time Series." MIT 1942. A war-time classified report published postwar. Cambridge, US-MA: MIT Press http://www.isss.org/lumwiener.htm.

ABOUT THE AUTHOR

[Editor: Author biography was current when the paper was initially published in 2017.]

William D. (Bill) Schindel is president of ICTT System Sciences. His engineering career began in mil/aero systems with IBM Federal Systems, included faculty service at Rose-Hulman Institute of Technology, and founding of three systems enterprises. Bill co-led a 2013 project on systems of innovation in the INCOSE System Science Working Group. He is an INCOSE Fellow, co-leads the patterns challenge team of the OMG/INCOSE MBSE initiative, and is a member of the lead team of the INCOSE agile systems engineering life cycle discovery project.

MISSOURI S&T

EARN YOUR SYSTEMS ENGINEERING DEGREE ONLINE

The flexibility you need. The depth your success depends on.

LEARN MORE

There's a reason Missouri University of Science and Technology leads the way in online systems engineering graduate education: We've been practicing and perfecting the art of using technology to engage and inspire for a long time. With faculty who teach based on real-world systems engineering experience, a worldwide network of successful alumni and a curriculum that sets standards across the industry, Missouri S&T is the perfect place for you.

https://online.mst.edu

INCOSE VOLUNTEER OPPORTUNITY

THE BEST ENGINEERS ALLOW FOR A LITTLE GIVE.

Become an INCOSE volunteer today!

incose.org/volunteer

INCOSE

A better world through a systems approach

CONNECT

Network and engage with the systems engineering community

LEARN

Enhance your knowledge through collaboration, research, and education

LEAD

Serve as an expert and thought leader to influence products and standards

PROSPER

Find career resources and improve your professional status

JOIN INCOSE TODAY

VISIT WWW.INCOSE.ORG/JOIN

INCOSE

International Council on Systems Engineering

A better world through a systems approach / www.incose.org

INCOSE Upcoming Events

Upcoming Events

Sep 2024
09
SWISSED24: Building Bridges
Zurich, Switzerland

Sep 2024
19–21
INCOSE's Annual Western States Regional Conference (WSRC)
Albuquerque, New Mexico, USA

Sep 2024
22–25
SESA's Systems Engineering Test & Evaluation (SETE) Conference
Melbourne, VIC, Australia

Sep 2024
23–24
Systems Analysis & Modelling (SAM) Conference
Linz, Austria

Nov 2024
05–06
INCOSE UK's Annual Systems Engineering Conference
Edinburgh, Scotland

Dec 2024
12–13
Complex Systems Design & Management (CSD&M) Conference
Paris, France

Feb 2025
01–04
INCOSE's Annual International Workship (IW)
Seville, Spain

Jul 2025
26–31
INCOSE's 35th Annual International Symposium (IS)
Ottawa, Canada

INCOSE