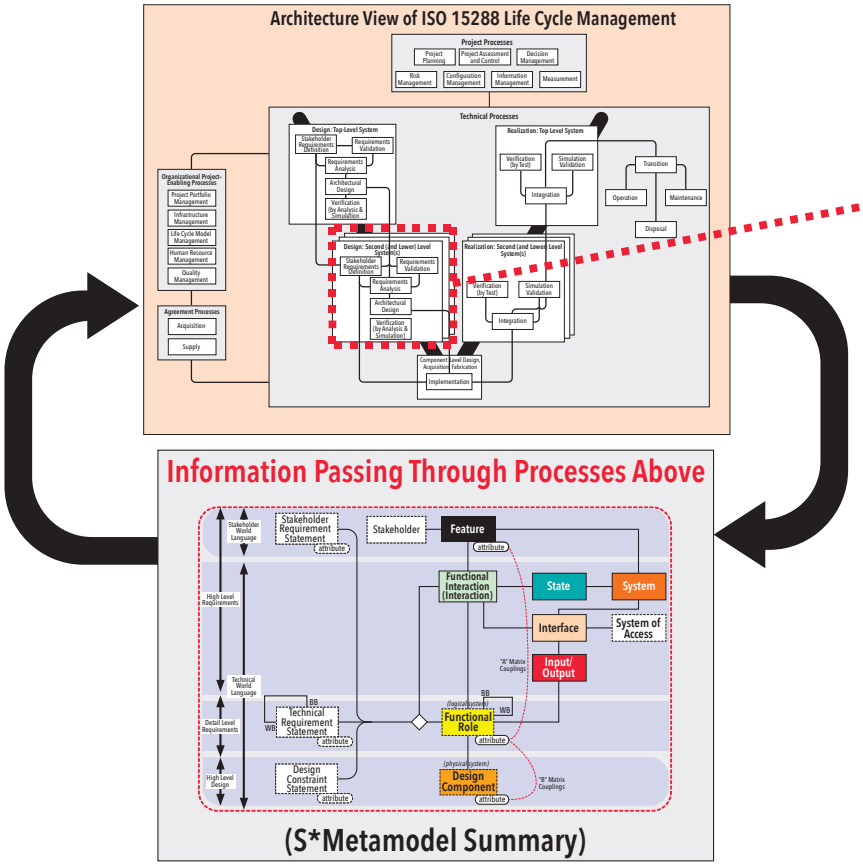
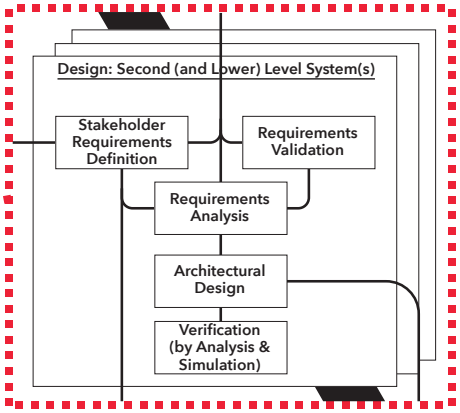
INSIGHT

This Issue's Feature:

Theoretical Foundations: Impacts on Practice II





Process versus information

Illustration credit: from the article *Maps or Itineraries? A Systems Engineering Insight from INSIGHT_Aug2024_vol 27-4, Ancient Navigators* by William D. Schindel

OCTOBER 2024
VOLUME 27 / ISSUE 5

A PUBLICATION OF THE INTERNATIONAL COUNCIL ON SYSTEMS ENGINEERING

MISSOURI S&T

Earn your M.S. in Systems Engineering online today!

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 global network of successful alumni and a curriculum that sets standards across the industry, Missouri S&T is the place for you.

online.mst.edu

Modular approach

No thesis required

30 credit hours

INSIGHT

A PUBLICATION OF THE INTERNATIONAL COUNCIL ON SYSTEMS ENGINEERING

OCTOBER 2024 VOLUME 27/ ISSUE 5

Inside this issue

About This Publication

INFORMATION ABOUT INCOSE

INCOSE's membership extends to over 25,000 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:

- 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
- October 2025 – 1 July 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 25,000 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 October Volume 27-5

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 Administration (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 South Alabama
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
Wichita State University College of Engineering
Woodward Inc
Worcester Polytechnic Institute (WPI)
Zuken, Inc

OCTOBER 2024
VOLUME 27/ ISSUE 5

FROM THE EDITOR-IN-CHIEF

William Miller, insight@incose.net

We are pleased to publish the October 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 as the state-of-the-art advances as evidenced in *Systems Engineering*, the Journal of INCOSE also published by Wiley, as well as papers presented at symposia and conferences by INCOSE and in the broader systems community.

The focus of this October issue of *INSIGHT* continues the systems engineering theoretical foundations and its impacts on practice in the August 2024 *INSIGHT* 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 for his sustained contributions in the MBSE Patterns Working Group to realize the Systems Engineering Vision 2035.

We lead the October *INSIGHT* with Bill Schindel's refreshing our understanding of the contributions to systems engineering by Ireland's Sir William Rowan Hamilton (1805-1865): "Innovation Ecosystem Dynamics: Value and Learning I: What Can Hamilton Tell Us?" Schindel states that Hamilton's profound contributions to science, technology, engineering, and math (STEM) deserve greater systems community attention. Supporting theory and practice, they remain relevant to the future of systems engineering (FuSE) initiative to realize the Systems Engineering Vision 2035. Key aspects apply to systems of all types, including socio-technical and information systems. Hamilton abstracted the energy-like generator of dynamics for all systems, while also generalizing momentum. Applied to the INCOSE innovation ecosystem pattern as dynamics of learning, development, and life cycle management, this suggests an architecture for integration of the digital thread and machine learning in innovation enterprises, along with foundations of systems engineering as a dynamical system.

"Realizing the Promise of Digital Engineering: Planning, Implementing, and Evolving the Ecosystem" elaborates on the benefits of digital engineering beyond implementing digital technologies. An ecosystem for innovation is a system of systems, only partly engineered, subject to risks and challenges of evolving socio-technical systems. This article summarizes an aid to planning, analyzing, implementing, and improving innovation ecosystems. Represented as a configurable model-based reference pattern used by collaborating

INCOSE working groups, it was initially applied in targeted INCOSE case studies, and subsequently elaborated and applied to diverse commercial and defense ecosystems. Explicating the recurrent theme of consistency management underlying all historical engineering, it is revealing of digital engineering's special promise, and enhances understanding of historical as well as future engineering and life cycle management. It includes preparation of human and technical resources to effectively consume and exploit digital information assets, not just create them, capability enhancements over incremental release trains, and evolutionary steering using feedback and group learning.

"Requirements Statements Are Transfer Functions: An Insight from Model-Based Systems Engineering" builds on traditional systems engineering paying attention to careful composition of prose requirements statements. Even so, prose appears less than what is needed to advance the art of systems engineering into a theoretically based engineering discipline comparable to electrical, mechanical, or chemical engineering. Prose requirements are subject to peoples' different impressions of their meaning. Model-based systems engineering might suggest the demise of prose requirements, but we argue otherwise. This article shows how prose requirements can be productively embedded in and a valued formal part of requirements models. This leads to the practice-impacting insight that requirements statements can be non-linear extensions of linear transfer functions, shows how their ambiguity can be further reduced using ordinary language, how their completeness or overlap more easily audited, and how they can be "understood" more completely by engineering tools.

"Feelings and Physics: Emotional, Psychological, and Other Soft Human Requirements, by Model-Based Systems Engineering" builds on traditional engineering encouraging requirements statements that are objective, testable, quantitative, atomic descriptions of system technical behavior. But what about "soft" requirements? When products deliver psychologically or emotionally based human experiences, subjective descriptions may frustrate engineers. This challenge is important for products appealing to senses of style, enjoyment, fulfillment, stimulation, power, safety, awareness, comfort, or similar emotional or psychological factors. Automobiles, buildings, consumer products, packaging, graphic user interfaces, airline passenger compartments and flight decks, and hospital equipment provide typical examples. This article shows how model-based systems engineering helps solve three related problems: (1) integrating models of "soft" human experience with hard technical product requirements, (2) describing how to score traditional "hard" technology products in terms of "fuzzier" business and competitive marketplace issues, and (3) coordinating marketing communication and promotion with the design process. The resulting framework integrates the diverse perspectives of engineers, stylists, industrial designers, human factors experts, and marketing professionals.

"Failure Analysis: Insights from Model-Based Systems Engineering" builds on system failure analyses such as failure mode and effects analysis (FMEA) as structured, well-documented, and supported by tools. Failure analyses can be perceived as (1) too labor intensive to encourage engagement, (2) somewhat arbitrary in identifying issues, (3) overly sensitive to the skills and background of the performing team, and (4) not building enough confidence of fully identifying the risks of system failure. This article shows how MBSE can answer these challenges by deeper and novel integration with requirements and design. Just as MBSE powered the requirements discovery process past its earlier, more subjective performance, so to can MBSE accelerate understanding and performance of failure risk analysis—as a discipline deeply connected within systems engineering.

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. ■


2025
Annual INCOSE
international workshop
HYBRID EVENT
Seville, SPAIN
February 1 - 4, 2025
MARK YOUR CALENDAR!
The INCOSE International
Workshop is traveling to
SEVILLE
https://www.incose.org/iw2025
INCOSE

OCTOBER 2024
VOLUME 27/ ISSUE 5

Innovation Ecosystem Dynamics, Value and Learning I: What Can Hamilton Tell Us?

William D. Schindel, schindel@ictt.com

■ **ABSTRACT**
Held in Dublin, Ireland, IS2024 invites us to refresh understanding of contributions to systems engineering by Ireland's greatest mathematician—Sir William Rowan Hamilton (1805–1865), professor of astronomy at Trinity College Dublin and royal astronomer of Ireland. His profound contributions to science, technology, engineering, and math (STEM) deserve greater systems community attention. Supporting theory and practice, they intersect foundations and applications streams of INCOSE's future of systems engineering (FuSE) program. Strikingly, key aspects apply to systems of all types, including socio-technical and information systems. Hamilton abstracted the energy-like generator of dynamics for all systems, while also generalizing momentum. Applied to the INCOSE innovation ecosystem pattern as dynamics of learning, development, and life cycle management, this suggests an architecture for integration of the digital thread and machine learning in innovation enterprises, along with foundations of systems engineering as a dynamical system.

■ **KEYWORDS:** Digital thread; Hamiltonian; Hamilton's principle; energy; momentum; machine learning; FuSE; future of systems engineering; foundations; organizational and social systems modeling.

INTRODUCTION

This paper highlights contributions William Rowan Hamilton made to the theoretical foundations of scientific and engineering disciplines, and some current questions to which they could apply. Hamilton's mathematically based patterns describe the phenomena of mechanics, electrical science, thermodynamics and subsequent disciplines, supporting the foundations of today's science, technology, engineering, and math (STEM). However, in the general setting of systems engineering, over-limiting assumptions about applicability sometimes arise. This paper briefly recalls aspects of Hamilton's contributions, and why current assumptions may be unnecessarily limiting practitioners. Prominent examples of current interest are noted—information systems and socio-technical systems of innovation. These suggest architecture-level strategies for integrating the digital thread and machine learning into the innovation enterprise. This is described in the perspective of the INCOSE innovation ecosystem pattern, which provides a general descriptive reference representation of enterprise or supply chain engineering and life cycle management processes, as a system of systems in its own right. This reference pattern interprets "innovation" very broadly, as including the entire life cycle of all products and systems, whether they are effective or not, providing a neutral framework for analysis use.

Millennia of observation and thought about natural phenomena were punctuated by a much shorter revolution. In less than 300 years, Newton, Lagrange, Gauss, Euler, Jacobi, Hamilton, Gibbs, and many others synthesized, extended, refined, and applied conceptual and mathematical frameworks that supported the dramatic acceleration of STEM. What followed rapidly changed the quality, length, and possibilities of human life.

Those mathematical frameworks provided conceptual and quantitative models to describe, predict, or explain many aspects of the modeled world, deterministic and probabilistic. Hamilton's contributions were recognized by later thought leaders as remarkably universal across phenomena of mechanics, electrical science, and other fronts. Max Planck (1858-1947) noted that "The chief law of physics, the pinnacle of the whole system is, in my opinion, the principle of least action"— Hamilton's principle (Planck 1925).

CONTEMPORARY INNOVATION ECOSYSTEM QUESTIONS

Systems engineering today frequently involves (1) information systems and (2) socio-technical systems. It is increasingly common for engineered products to directly involve these domains, and even more common for the engineering enterprise itself to depend upon them. Even though they were not the main interest in Hamilton's time, today these domains have rapidly growing significance for systems engineers.

Related engineering project questions that Hamilton's contributions may help us answer include:

A. **Project and program planning:** What are predicable efforts, times, and costs of performing innovation and life cycle management? What are related uncertainties (and consequent risks) in those predictions? These are questions addressed historically by empirical models such as COSYSMO (Valerdi, Boehm, and Reifer 2003) and more recently asked by INCOSE FuSE foundations efforts (de Weck 2023). They are supported by basic shared understandings of enterprise processes such as ISO (2023) and Walden et al. (2023). When projects involve complications of organization (such as supply chains or consortia), problems with communication, incentives, shared understandings, or cultures, their success may be doomed before execution begins.

B. **Project execution management:** As projects are performed (and encounter real-world perturbations only partly predictable), what are the means of preparing, monitoring, and directing them for optimum outcome—including decision-making in particular? During complex multi-enterprise development projects, how can we detect and act on systemic project uncertainties and instabilities threatening success? These are questions addressed historically by disciplines such as capabilities assessment (SEI 2010), project management in general (Rebentisch 2017), agile methods in particular (Dove 2001), and emerging aspects of digital engineering (Schindel 2022).

C. **Project learning and its recurrent application:** What are means and effects of accumulating new experience in items (A) and (B) above, and effectively distilling, managing trust in, and applying knowledge and competency in future projects? This question is addressed historically by technical readiness levels (Mihaly 2017), capability maturity models (SEI 2010), knowledge management (Trees, McCulloch, and Witt 2021), application of recurrent patterns (Alexander 1977), Gamma et al. 1994, Cloutier 2008, and Schindel 2022), and product line engineering (Clements and Northrop 2002 and ISO 2021). It includes the emerging subject of machine learning (LeCun, Bengio, and Hinton 2015).

D. **Information and information system roles in items (A), (B), and (C) above:** A common thread through the above are roles of information and information systems—both those using engineered information technologies and those performed by human beings. What is the theoretical basis for engineering the performance of these subsystems as an integrated part of the larger enterprise systems in which they appear? How can systems engineering connect these? These questions are addressed historically by information theory (Shannon 1948), enterprise architecture (Foorthuis, Steenbergen, Brinkkemper, and Bruls 2016), digital engineering (Schindel 2022), the digital thread (Cribb et al. 2023), and machine learning (LeCun, Bengio, and Hinton 2015). More recently, US and European governments are issuing executive orders and regulations demanding new levels of mastery of what is emerging.

What can Hamilton tell us about the above questions?

A CHALLENGE TO CONTEMPORARY ASSUMPTIONS

Hamilton's framework may be most familiar to engineers in mechanical, civil, or electromagnetics settings. The systems community may be assuming that Hamilton's mathematical contributions do not address the socio-technical and information system questions above in a practical way.

One sign of such an assumption in the INCOSE and other systems communities is a continued call and search for what are perceived as missing theoretical foundations for the science and engineering of generalized systems (Friedenthal et al. 2021). Disciplines in engineering and sciences are concerned with phenomena (e.g., mechanical, electrical, and chemical) specific to those disciplines, leading to impactful phenomena-specific patterns of interactions described by laws specific to those disciplines, often in mathematical form. What about equivalent impactful phenomena, theory, and mathematics for systems in general?

A counterargument is that more attention should be given to already modeled phenomena (from Hamilton and other STEM pioneers) before spending too much effort looking elsewhere (Schindel 2016 and 2020). Three such phenomena have been suggested, playing parts in this paper: (1) the system phenomenon, studied by Hamilton; (2) the value selection phenomenon, fueling innovation force; and (3) the group learning and trust phenomenon, learning and applying patterns in the face of uncertainties.

We do not suggest that unnoticed phenomena and laws concerning information systems and socio-technical systems are not waiting for discovery. However, as already noted by those who followed him, Hamilton's framework is not limited to only mechanical or other specific phenomena.

INFORMAL SUMMARY OF THE SYSTEM PERSPECTIVE INFORMED BY HAMILTON

Hamilton (1834) showed we can describe energy (or at least an energy-like characteristic function) of a system in a general and mathematical way not restricted to only some systems. Hamilton and those who followed showed how deeply these concepts follow from the most limited set of ideas present in many systems—even seemingly "soft" systems. Only the concepts of system interaction and state are required to get started. An informal argument proceeds as follows:

A. **Systems:** Start with a system *of* any type. By "system", we mean a set of interacting system components (Figure 1). By "interact" we mean they exchange input-outputs, such as force, material, energy, or information, resulting in changes of state of the components. By "state" of a component we mean the condition of the component that can modify its current input-output behavior. Interaction thus changes state, which in turn impacts interaction.

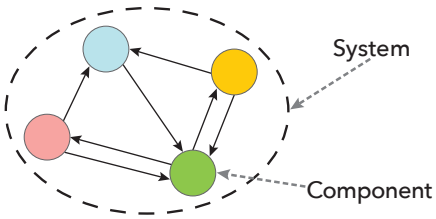


Figure 1. The system perspective

B. **Non-Deterministic and Discrete Systems:** This short and informal discussion focuses on deterministic, smoothly continuous systems. However, it turns out that discrete Hamiltonian systems have been heavily explored and exploited, including providing the symplectic Hamiltonian integrators found in numerical simulation (Shibberu 1994, and Marsden and West 2001). For

non-deterministic cases, Hamiltonian mechanics provide the foundations of the rich historical field of statistical mechanics (Gibbs 1901 and Khinchin 1949), where state flows are replaced by probability density flows. Probabilistic cases also re-enter this story through machine learning and human behavior.

C. **States:** Have a way of representing the state of the system of interest $Q(t) = \{q_1, \ldots, q_n\}$, whose values change over time at rate $\dot{Q}(t) = \{\dot{q}_1, \ldots, \dot{q}_n\}$, believed sufficient to characterize observed interactions.

D. **Characterizing System Level Behavior:** Imagine now a scalar-valued function of state and time, *not yet defined until below*, contributed by Hamilton: $H(Q, \dot{Q}, t)$, intended to characterize something about the system—we have not said how yet.

E. **Generalized Momentum:** Hamilton contributed a "generalized momentum," $P(t) = \{p_1, \ldots, p_n\}$, intended to generalize the idea of momentum in elementary physics — describing ability to change $\dot{Q}$. His generalized P is defined by the sensitivity of H (H not otherwise defined yet) to $\dot{Q}(t)$:

$$p_i \equiv \frac{\partial H}{\partial \dot{q}_i} \tag{1}$$

(Notice that if H turns out to be something "like" energy, this says that momentum is the sensitivity of energy to changes in velocity, or that energy is required to change velocity, an intuitively reasonable generalization of mechanical systems.)

F. **Defining the Hamiltonian:** We want H to characterize the system's (Q, P) trajectories, and will do so here by tying them to the local slopes of surface H. (See Figure 2.) First, the local sensitivity of H with respect to p_i at (Q, P) is to be equal to the time rate of change of q_i along the system state trajectory passing through (Q, P):

$$\dot{q}_i = \frac{\partial H}{\partial p_i} \tag{2}$$

Second, the local sensitivity of H with respect to q_i at (Q, P) is to be the negative of the time rate of change of p_i along the system state trajectory passing through (Q, P):

$$\dot{p}_i = -\frac{\partial H}{\partial q_i} \tag{3}$$

For intuition, notice that dividing both sides of Eq (2) by both sides of Eq (3) shows that the instantaneous direction of motion in the (q, p) plane of Figure 2 is the same as the ratio of the local slopes of H in the q and p directions.

The above reasoning is important to intuitive motivation and perspective on applying Hamiltonians. Hamilton took the major step of providing Eq (1) as a definition of generalized momentum, but defined H through a Legendre transformation of a pre-existing Lagrangian, which we are not assuming here, as we are making no assumption of pre-existing energy concepts. A traditional textbook perspective is to start with a *mechanical* system having defined kinetic and potential energies and a Lagrangian, then applying a Legendre transformation to yield a Hamiltonian that is based conceptually and mathematically on mechanical energy (Greenwood 1977, and Landau and Lifshitz 1976). In that reasoning path, one then proves that Eq (2) and (3) follow. Here, we instead define H as a function satisfying Eqs (1), (2), and (3), for a collection of actual trajectories, whether known or unknown. The mathematical question of existence (not all systems are Hamiltonian) is informally addressed in item H that follows.

G. **Hamilton's Equations:** Equations (2) and (3) are Hamilton's equations for the time evolution of the state of the system—they are stated as equations of motion, describing trajectories in terms of H. It may seem odd that we have arrived at the equations of motion of a system, but we do not know what specific kind of system it is yet! Intuitively, this is because we started with a set of trajectories and invented a real-valued function of state that characterizes those trajectories. See Figure 2.

H. **A "Story Experiment":** To ground ourselves in both intuitive and practical framing of Hamilton's equations, here is a related "story" experiment that in recent years has been repeatedly performed by multiple parties for different types of systems, with variant approaches including Bertalan (2019), Greydanus (2019), Toth and Rezende et al. (2020), Bhat (2020), and Chen and Tao (2021):

i. Identify a specific system of interest, of any type, that you can directly observe.

ii. As the system operates, observe and record a series of $(Q, \dot{Q}, t)$ state trajectory tuple samples.

iii. Set up a machine learning (ML) system to "learn" (discover) a functional surface $H(Q, \dot{Q}, t)$ that minimizes across the sample space the following learning loss functional:

$$\text{Loss}[H(Q, \dot{Q}, t] = \sum \{(q_i - \frac{\partial H}{\partial p_i})^2 + (\dot{p}_i + \frac{\partial H}{\partial q_i})^2 + (p_i - \frac{\partial H}{\partial q_i})^2\} \tag{4}$$

iv. Two terms of Eq (4) show the "learned surface" attempts to satisfy Hamilton's Equations (2) and (3) for the observed training data. The third term attempts to satisfy (1) to discover generalized momentum. Thus, we can "discover" a Hamiltonian surface from observational data.

The main point of this "story experiment" is not machine learning—it is that Hamilton provided a conceptual function H that *characterizes the dynamic behavior of any system having deterministic continuous state trajectories* (see also B, F above), by how the function H defines a "map" of state trajectories. H is a characterization of the system, sometimes referred to as a "generator" of the system's dynamics. In any neighborhood of the (Q, P) plane where we have a set of observations, those observations can provide estimated time rates of change for Q and P along the system's state trajectory. From that, Hamilton's equations effectively define H quantitatively (up to an additive constant) by telling us about the local slopes in the surface of H at those points with respect to Q and P axes. See Figure 2(a). Knowing nothing except the observed trajectories of the system, we have created the surface H, which thereby characterizes that system's behavior (as trajectories in the Q, P plane).

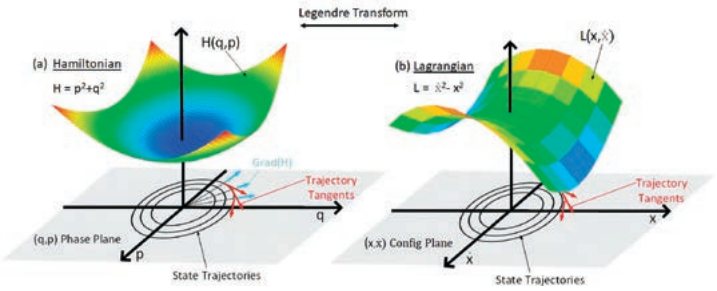


Figure 2. Phase plane, Hamiltonian; config plane, Lagrangian—for simple harmonic oscillator

I. **The Variational Version:** In Figure 2(a) a trajectory of the system occurs in the (Q, P) phase plane. At a given point on that trajectory, its direction in that plane is the tangent vector $\{\dot{Q}, \dot{P}\}$. The gradient of H in the (Q, P) phase plane is the vector $\{\frac{\partial H}{\partial Q}, \frac{\partial H}{\partial P}\}$, pointing in the direction of maximum rate of change of the surface H(Q, P). In that plane, and perpendicular to the gradient, is vector $\{\frac{\partial H}{\partial P}, -\frac{\partial H}{\partial Q}\}$, pointing in the direction of zero rate of change (constancy) of the surface H(Q, P). But based on Hamilton's equations above, that is the same as the trajectory tangent vector, $\{\dot{Q}, \dot{P}\}$. So, the trajectory of the system moves in the direction of zero rate of change of the surface H. H is thus invariant (constant, conserved) in time along its trajectory, by the very definition of H. Figure 2(b) shows the Lagrangian surface L for the same system. It expresses the variational statement of Hamilton's principle (Lanczos 1986), noted by Max Planck as remarkably broad. Here, we see that it can apply to many systems for which we can define states, including information systems and socio-technical systems.

J. **Holonomic? Conservative?:** The proposed potential energy concepts described in the next (application) section suggest that the systems of interest described there for Hamiltonian treatment are holonomic. In the main dynamics implied for such information and socio-technical systems, H there appears possibly conserved, whether or not it is called "energy." See also "dissipation" later below.

APPLICATION: SYSTEM STATES AND LEARNING IN AN INNOVATION ECOSYSTEM

The above discusses the dynamic evolution of system state variables Q(t). But what are the practical, real project state variables that we care about for the enterprise information and innovation project questions listed earlier above? The following sections focus on some key state variables.

Ecosystem States Associated with Learning

The American Institute of Aeronautics and Astronautics published AIAA's Digital Thread Reference Model (Cribb et al. 2023). The core of this AIAA reference model is based on the INCOSE agile systems engineering life cycle management (ASELCM) pattern (also known as the innovation ecosystem pattern) (Schindel and Dove 2016). A central theme of these reference models is the paradigm of "consistency management" (Schindel 2021), which seeks to manage over the duration of a project the reduction (ideally, to zero) of a set of managed consistency "gaps" that are familiar in the history of engineering and life cycle management projects, and which run through the backbone processes of ISO (2023) and Walden et al. (2023). A few prominent examples of the long list of consistency issues are:

- Is the product design consistent with its requirements?
- Are those requirements consistent with the mission and stakeholder needs and priorities?
- Are the emergent behaviors (both required and to be avoided) in the engineered system consistent with the experience about the underlying phenomena from which they emerge?
- Are instances of the manufactured product consistent with the design specifications?
- Is the observed use of the product consistent with the product mission and requirements?
- Is performance of the deployed product consistent with the specified requirements?
- Is the environment of use of the product consistent with its representation in the product mission and requirements?
- Reducing these and other consistency gaps generates learned information. Learning occurs over the course of projects, much of it by humans, with some of it captured in artifacts and some in tribal knowledge. In current and future projects, more of this learning includes digital engineering agency that is only partly human, with more learning captured in digital artifacts.

The earlier list of project questions and the above consistency management paradigm now help us see a project as two kinds of mathematical boundary value problems:

- **Boundary Value Problem 1—The web of end-state consistencies:** Figure 3 illustrates the idea that a product design, implemented, delivered and in service, reflects selection pressures to minimize a set of consistency gaps. Visualize equilibrium "relaxation" of the springs into their "trade off" positions during a project. Project time is not explicitly visible in this view, although some of the consistencies it shows may themselves be about project time.

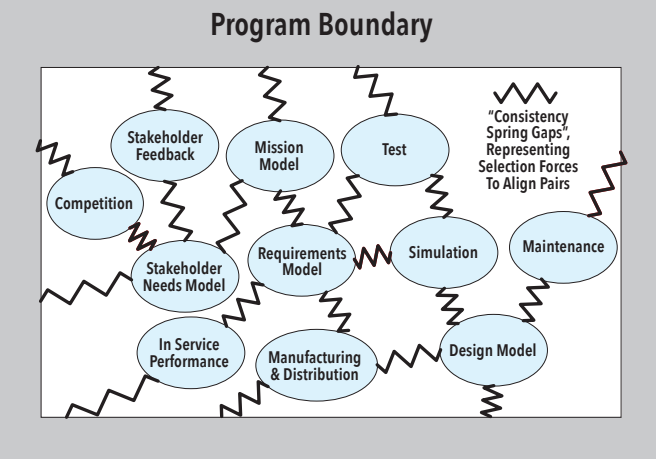
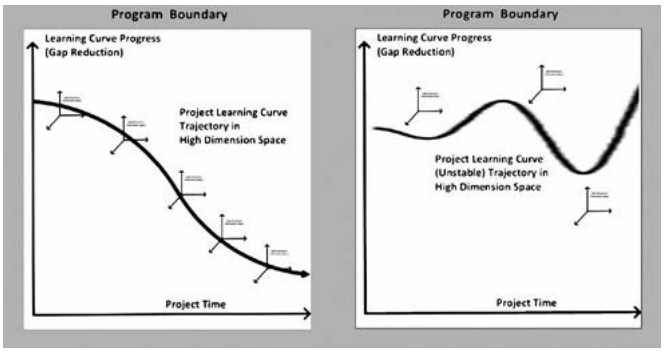


Figure 3. Example consistency gap web of elastic springs—a metaphor

- **Boundary Value Problem 2—The dynamics of state evolution over the project duration:** In contrast to that end state view, Figure 4(a) illustrates the idea of what occurs during a project, as a dynamic trajectory in high dimension space, progressing over time. (One might visualize the elastic network of Figure 3 "vibrating" and "relaxing" during this time.) Analogous to a control system boundary value problem, this perspective is more about questions concerning the dynamic behavior of the project itself over time, as a dynamical system.



4(a) Well behaved learning *4(b) Ringing, unstable*

Figure 4. Example learning curve trajectories for a project

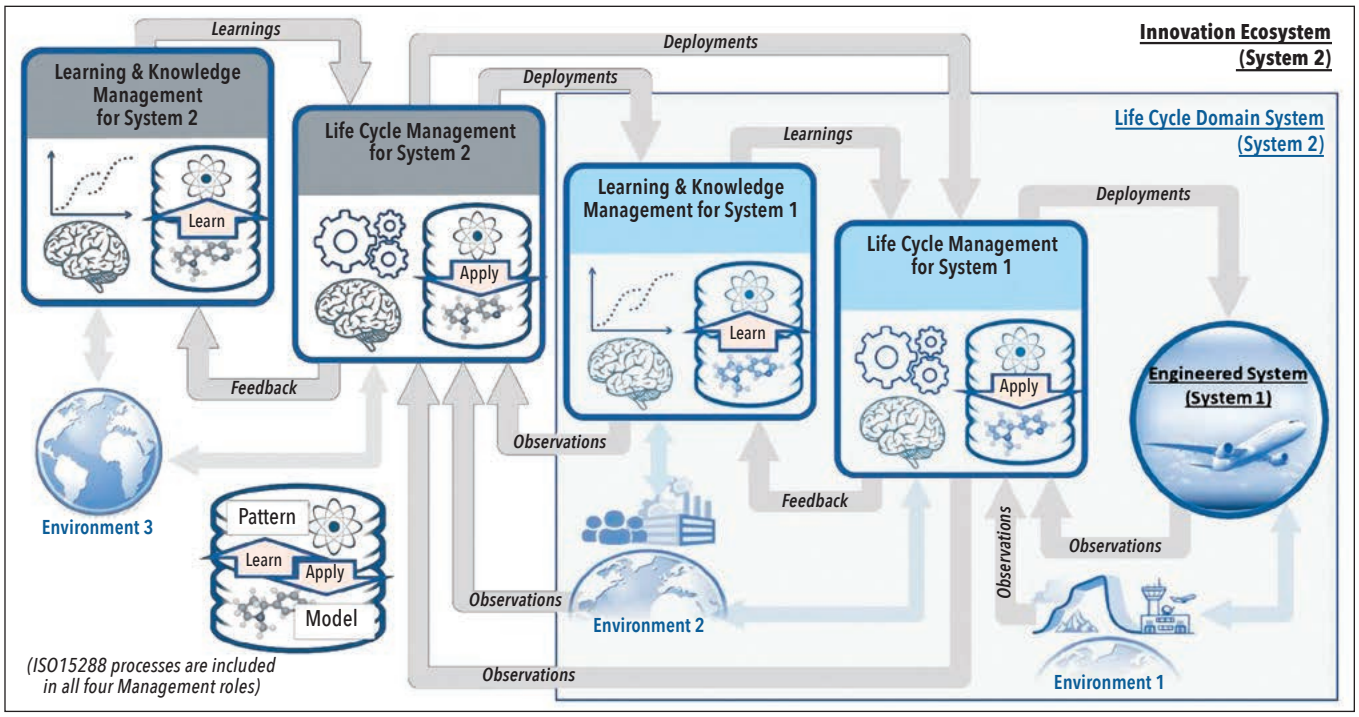


Figure 5. ASELCM Pattern, Level 1 view, separating learning from application (adapted from Cribb (2023))

We are not guaranteed that an actual dynamic project system will be well-behaved, converging to a deliverable good outcome. It may "ring" or become unstable, illustrated by Figure 4(b). Hamilton's contribution of "generalized momentum" (discussed in the previous section) ultimately figures into this. Figures 3-4 illustrate that, for a development and life cycle management project, it is the states of the managed consistency gaps that we should especially care about as candidates for the ecosystem state trajectory model in the Hamilton perspective. Further, this trajectory can be seen as the innovation ecosystem learning the information necessary to reduce inconsistencies to deliverable levels.

This also prepares us to differentiate between what was already known at the start of the project (*a priori* knowledge; "priors"), versus what is learned during the project. That differentiation is central to the practical integration and management of learned formal patterns expressed as parameterized models, along with more informal tribal knowledge and heuristics. It is also central to application of Bayesian inference (Jaynes, 2003), dramatically successful in communication and navigation systems.

The Level 1 (Figure 5) view of the ASELCM pattern (Schindel and Dove 2016, and Schindel 2022) incorporates that differentiation, showing:

1. Life cycle management for System 1 acts based on *what is already known* about System 1 and its environment;
2. Learning and knowledge management for System 1, for learning new information about System 1 and its environment (whether human-based learning, machine learning, or their combination).

The ASELCM level 2 (Fig. 6) view shows:

1. the already learned deployed generic model (pattern), more general than needed for the specific project, hence to be configured;
2. the configured specific model, specific for the project.

This reference model is not to say that human enterprise project teams always respond optimally, but rather to study the forces to which they respond, by representing the perceived loss functions. These can also be central to automated machine learning algorithms.

Machine Learning in the Ecosystem

Public awareness of machine learning progress has recently grown dramatically. However, it results from 75 years of efforts across dramatically improving methods, along with orders of magnitude advance

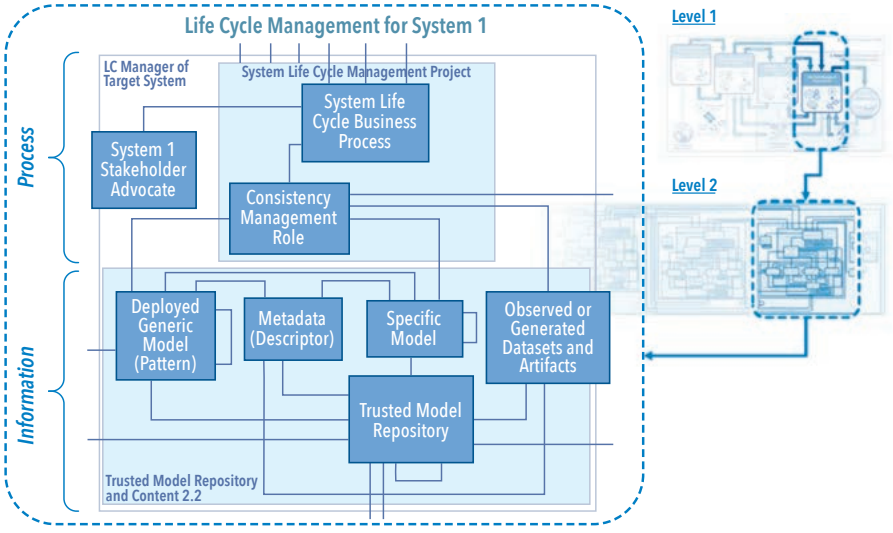


Figure 6. ASELCM pattern, level 2 view, information versus processes (adapted from Cribb (2023))

in hardware and training data resources (LeCun, Bengio, and Hinton 2015).

Central to the contemporary machine learning work is the concept of minimization of some form of loss function by various training algorithms. Hopfield's seminal PNAS paper of 1982 (Hopfield 1982), reawakening artificial neural network interests, described minimization of what he referred to as an "energy" function. Inspired at the time by properties of both biological neural networks and physics of dynamical system state flow patterns, Hopfield referred to results as "isomorphic with an Ising model" of physics.

In the more recent efforts (LeCunn 2006, 2021), "energy-based methods" have become popular in machine learning. The continued reference to "energy" in this work stems from recognition of the deep connection between probability distributions governing the performance of neural nets over large sample spaces and the probability distributions of statistical physics (for example, Gibbs-Boltzmann distribution; Helmholtz free energy distribution, and Hinton and Zemel 1993).

The State Variables

As illustrated above, the key "project state variables" we want to manage effectively (possibly with help from Hamilton) include the consistency gap signals. These contribute to the "potential energy" (Q related) part of the Hamiltonian, as they describe the "consistency gap field" that any project seeks to minimize through selection forces. However, these are not the only state variables, as the metaphor of Figure 3 is replaced by inter-role selection force interactions in Figure 6; its consistency management roles and business process roles contribute additional state variables further characterizing the organization's processes, capabilities, and culture.

REAL PROJECTS: DECISION-MAKING AND THE DIGITAL THREAD

Executing a project involves *making decisions*. Some of these decisions are high-visibility major choices by senior decision-makers, at major stage gates. Many other decisions occur across the teams on a day-today basis. With the above consistency management paradigm, we can think of those decisions across the system life cycle in an additional way: *All program decisions are reconciliations of inconsistencies* (Schindel 2023, 2024).

The digital thread reference model (Cribb et al. 2023) represents roles of detection of inconsistencies (by human or automated agents) and reconciliation of those inconsistencies (more likely by human agents, potentially with future automated assistance). Snapshot records of the related information items form a "consistency thread" precursor of the digital thread of Figure 7.

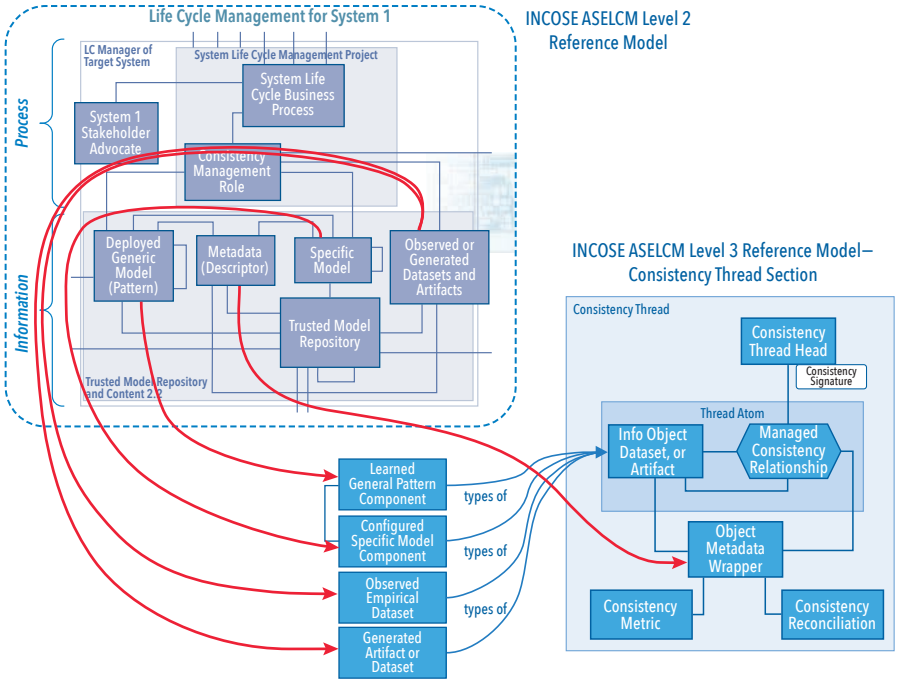


Figure 7. ASELCM pattern, consistency thread view (adapted from Cribb (2023))

In the innovation system dynamics, the resulting consistency thread/digital thread plays these major roles:

1. From a system dynamics perspective, it is a trace of the project state trajectory of Figs 3 and 4.
2. It is also the record of detected inconsistencies, and their reconciliations.
3. It exposes data for use in learning. Whatever the project outcome, it provides a learning database, for human or machine learning.
4. It provides support for the use of past learning.

Figures 5, 6, and 7 summarize aspects of the architectural pattern for integration of the enterprise, the digital thread, and human and machine learning.

Are Classical Physics Models Practical for Socio-Technical Systems?

The question of treating execution of complex, risky innovation projects as a mathematical problem of optimal control was considered in Schindel (2017). However, it is reasonable to question whether using Hamilton's mathematically based model is practically plausible for complex, human-performed sociotechnical systems such as engineering and life cycle management. Do differential equations really have any practical place here? Similarly, is it reasonable to expect that machine learning can be productive in this human judgment-intensive technical context?

That such questions would even be seriously considered has recently become more likely, based on advancements leading to surprising demonstrations, such as machine learning informed algorithms passing legal bar and medical licensing examinations or performing diagnoses. How is it that a machine learning algorithm based on Jacobian matrices of partial derivatives and flowing with numbers has led to such capabilities? While the answers are emerging, clearly earlier intuition about the limitations of mathematics of classical mechanics in this space needs to be recalibrated now, because of demonstrated progress in performance enabled by better algorithms, accessible training data, and hardware capacities.

At the very least, this encourages preparatory re-acquaintance with Hamilton's pattern. Even currently human-intensive cases begin to illustrate the enterprise architecture into which advanced versions of the digital thread can be integrated for enterprise learning. The discrete time and statistically-based versions of Hamilton's pattern are likely to be the most relevant for the innovation ecosystem—but that is already the case for much of contemporary engineering's use of Hamilton's contributions.

If we hope to apply the methods of optimal estimation and control in the presence of randomness and uncertainty (they have been very successful for simpler

engineered systems) to the system of engineering and life cycle management itself (Schindel 2017), then we first need to have a theoretical representation of that system. Likewise, if we want to have a theoretical basis for understanding the behavior of autonomous learning and inference algorithms of artificial intelligence (Cribb et al. 2023), then we need sufficient representation of them as dynamical systems. It appears that Hamilton and those who followed have provided us with such a representation, if we reason in the right order.

Ecosystem Selection Forces, Dissipation, Entropy, and Complexity

The above application discussion focused on potential energy in the innovation ecosystem, but the selection forces provided by other ecosystem roles (Schindel 2020, 2023, 2024) contribute kinetics to the dynamical behavior of this system. For discussion in a subsequent paper, certain aspects beckon:

1. Dissipation is about reversibility. As learning proceeds in an innovation ecosystem project, the potential energy associated with consistency gaps shrinks macroscopically at the ecosystem level. *If* the ecosystem is to conserve H, what (kinetic, potential) would grow to offset that shrinkage? An interesting candidate is the project's digital thread information, captured during learning to "explain" (and defend for posterity) a learned product model's validity as a compression of empirical data. At a more microscopic level, Landauer, Bennet, Feynman, Toffoli, and others have pursued the concepts of dissipation-free information processing, with the exception of dissipation by erasure. (Hey 1996).
2. Hamiltonian systems also conserve information entropy (Carcassi and Aidala 2020). Using Kolmogorov definition of complexity as size of the generator (Li and Vitany 1997), and recalling Shannon entropy's connection to encoded message size, may imply a form of conservation of complexity of the engineered system in the ecosystem (de Weck 2023). However, note that complexity of what the ecosystem of Figure 5 must learn in a project is not the full complexity of the engineered System, but of the "posterior" aspects of it—separating "what do we *already* know?"
3. The learning subsystem of the ecosystem can be Hamiltonian (Ramacher 1992)
4. In the study of dynamical systems, a long and rather complex history of research dating to Hertz in 1894 has described the nature and consequences of non-holonomic constraints. (Bloch 2003, Rojo and Bloch 2018, Flannery 2005, and Eden 1951).

CONCLUSIONS AND FUTURE WORK

This synthesis paper has:

1. Outlined some of the strategic questions faced by contemporary innovation ecosystem projects;
2. Provided an informal refresher on how Hamilton's framework can apply to diverse systems, including socio-technical and information systems, and to the innovation ecosystem in particular;
3. Shown that the key innovation ecosystem state variables relevant for Hamiltonian "potential" modeling include the ASELCM pattern consistency management "gaps" central to the digital thread;
4. Noted that energy based learning methods for machine learning algorithms are already being used to learn real system Hamiltonians as well as being Hamiltonian modeled themselves;
5. Shown that consistency management's needs for inconsistency detection and reconciliation are candidates for machine learning based aids to traditional labor-intense roles;
6. Shown that this synthesis suggests an innovation enterprise architecture integrating the digital thread as well as machine and human learning;
7. Laid a foundation for future momentum kinetics and applications work utilizing these approaches, as well as case study work.

Related work continues to progress in the INCOSE Patterns Working Group, supporting the FuSE initiative, and additional collaborations with other working groups, societies, and enterprises. ■

ACKNOWLEDGEMENTS

We have to thank W. R. Hamilton for the core mathematical insights described here—even if we must rediscover them. Rick Dove led the INCOSE agile systems engineering discovery project during which the ASELCM pattern described here was applied and advanced. When the ASELCM pattern was used as the basis of the AIAA aerospace digital thread reference model, Woong Je Sung contributed the friendlier but still correct diagrams of the original SysML pattern shown here as Figures 5, 6, and 7, for general audiences.

REFERENCES

- Alexander, C. 1977. *A Pattern Language: Towns/Buildings/Construction*. Oxford University Press.
- Bertalan, T., et al. 2019. "On Learning Hamiltonian Systems from Data." *Chaos* 29: 121107-1-9. AIP Publishing.
- Bhat, H., K. Ranka, and C. Isborn. 2020. "Machine Learning a Molecular Hamiltonian for Predicting Electron Dynamics." *Intl J of Dynamics and Control* 8: 1089–1101.
- Bloch, A. 2003. *Nonholonomic Mechanics and Control*. New York, US-NY: Springer.
- Carcassi, G., and C. Aidala. 2020. "Hamiltonian Mechanics is Conservative of Information Entropy." *Studies in History and Philosophy of Science, Part B: Studies in History and Philosophy of Modern Physics*, August, 60-71.
- Chen, R., and M. Tao. 2021. "Data-Driven Prediction of General Hamiltonian Dynamics via Learning Exactly-Symplectic Maps." 38th International Conference on Machine Learning 139:17171727.
- Clements, P. and L. Northrop. 2002. *Software Product Lines: Practices and Patterns*. Addison-Wesley.
- Cloutier, R. 2008. *Applicability of Patterns to Architecting Complex Systems: Making Implicit Knowledge Explicit*. Saarbrucken, DE: VDM Publishers.
- Cribb, M., et al. 2023. "Digital Thread: Definition, Value, and Reference Model." American Institute of Aeronautics and Astronautics.
- de Weck, O. 2023. "The First Law of Systems Science: Conservation of Complexity." INCOSE 2023 International Workshop, Los Angeles, US-CA.
- Dove, R. 2001. *Response Ability: The Language, Structure, and Culture of the Agile Enterprise*. Wiley. ISBN-10: 9780471350187.
- Eden R. 1951. "The Hamiltonian Dynamics of Non-Holonomic Systems." *Proc. of the Royal Society*, 07 March 205 (1083). DOI:https://doi.org/10.1098/rspa.

- Flannery, M. 2005. "The Enigma of Nonholonomic Constraints." *Am. J. Phys.* 73 (3).
- Foorthuis, R., M. Steenbergen, S. Brinkkemper, and W. Bruls. 2016. "A Theory Building Study of Enterprise Architecture Practices and Benefits." *Information Systems Frontiers* 18 (3): 541564. DOI: 10.1007/s10796-014-9542-1.
- Friedenthal, S., et al. 2021. *INCOSE Systems Engineering Vision 2035: Engineering Solutions for a Better World*. International Council on Systems Engineering, San Diego, US-CA.
- Gamma, E., et al. 1994. *Design Patterns: Elements of Reusable Object-Oriented Software*. AddisonWesley.
- Gibbs, J. W. 1901. *Elementary Principles of Statistical Mechanics*. Garden City, US-NY: Dover Publishers.
- Greenwood, D. 1977. *Classical Dynamics*. Mineola, US-NY: Dover Publications.
- Greydanus, S., et al. 2019. "Hamiltonian Neural Networks." Proc. of NeurIPS 2019, Vancouver, CA-BC.
- Hamilton, W. 1834. "On a General Method in Dynamics." *Phil. Trans. of the Royal Society, Part II* 1834: 247-308.
- Hey, A., ed. 1996. *Feynman and Computation: Exploring the Limits of Computers*. Cambridge, US-MA: Perseus.
- Hinton, G., and R. Zemel. 1993. "Autoencoders, Minimum Description Length and Helmholtz Free Energy." Proc of Advances in Neural Information Processing Systems 6 (NIPS 1993): 3-10.
- Hopfield, J. 1982. "Neural Networks and Physical Systems with Emergent Collective Computational Abilities." *Proc. Natl. Acad. Sci. US* 79: 2554-2558, Biophysics.
- ISO. 2021. ISO/IEC 26580:2021, Software and Systems Engineering: Methods and Tools for the Feature-Based Approach to Software and Systems Product Line Engineering. Technical Committee ISO/IEC JTC 1/SC 7. ICS: 35.080.
- ISO. 2023. ISO/IEC/IEEE International Standard–Systems and Software Engineering — System Life Cycle Processes. ISO/IEC/IEEE 15288:2023, doi: 10.1109/IEEESTD.2023.7106435.
- Jaynes, E. 2003. *Probability Theory: The Logic of Science*. Cambridge University Press. ISBN-10: 0521592712.
- Khinchin, A. 1949. *Mathematical Foundations of Statistical Mechanics*. New York, US-NY: Dover Publishers.
- Lanczos, C. 1986. *The Variational Principles of Mechanics*, 4th Edition. New York, US-NY: Dover Publishers.
- Landau, L, and E. Lifshitz. 1976. *Mechanics*, Third Edition. London, GB: Butterworth Heinemann.
- LeCun, Y. 2021. The Energy-Based Learning Model. Lecture video of May 18. https://www.youtube.com/watch?v=4lthJd3D-NTM.
- LeCun, Y., Y. Bengio, and G. Hinton. 2015. "Deep Learning." *Nature* 521: 436-444, MacMillan.
- LeCun, Y., et al. 2006. "A Tutorial on Energy-Based Learning." *Predicting Structured Data*, Cambridge, US-MA: MIT Press.
- Li, M., and P. Vitany. 1997. *An Introduction to Kolmogorov Complexity and its Applications*, Second Edition. Springer.
- Marsden, J., and M. West. 2001. "Discrete Mechanics and Variational Integrators." *Acta Numerica* 357-514.
- Mihaly, H. 2017. "From NASA to EU: The Evolution of the TRL Scale in Public Sector Innovation." *The Innovation Journal* 22: 1–23.
- Planck, M. 1925. *A Survey of Physics: A Collection of Lectures and Essays*, Transl. by R. Jones and D. H. Williams, Methuen & Co., Ltd.
- Ramacher, U. 1993. "Hamiltonian Dynamics of Neural Networks." *Neural Networks* 6: 547-557.
- Rebentisch, E., editor. 2017. *Integrating Program Management and Systems Engineering: Methods, Tools, and Organizational Systems for Improving Performance*, Hoboken, US-NJ: Wiley. ISBN-10:9781119258926.
- Rojo, A., and A. Bloch. 2018. *The History and Physics of the Least Action Principle*. New York, US-NY: Cambridge University Press.
- Schindel, W., and R. Dove. 2016. "Introduction to the Agile Systems Engineering Life Cycle MBSE Pattern." INCOSE 2016 International Symposium, Edinburgh, GB-SCT.
- Schindel, W. 2016. "Got Phenomena? Science-Based Disciplines for Emerging Systems Challenges." INCOSE 2016 International Symposium, Edinburgh, GB-SCT 26 (1): 2256–2271.
- ———. 2017. "Innovation, Risk, Agility, and Learning, Viewed as Optimal Control & Estimation." INCOSE 2017 International Symposium, Adelaide, AU.
- ———. 2020. SE Foundation Elements: Discussion Inputs to INCOSE Vision 2035 Theoretical Foundations Section,V2.3.2a. INCOSE Patterns Working Group. Download from: https://www.omgwiki.org/MBSE/lib/exe/fetch.php?media=mbse:patterns:-science_math_foundations_for_systems_and_systems_engineering—1_hr_awareness_v2.3.2a.pdf.
- ———. 2021. Consistency Management as an Integrating Paradigm for Digital Life Cycle Management with Learning, INCOSE MBSE Patterns Working Group, download from—https://www.omgwiki.org/MBSE/lib/exe/fetch.php?media=mbse:patterns:aselcm_pattern_—_consistency_management_as_a_digital_life_cycle_management_paradigm_v1.3.1.pdf.
- ———. 2022. "Realizing the Value Promise of Digital Engineering: Planning, Implementing, and Evolving the Ecosystem." *INSIGHT Special Issue on Digital Engineering* 25 (1).
- ———. 2023. All Decisions Across Life Cycles of Systems are Reconciliations of Inconsistencies. Presentation to INCOSE North Texas Chapter, Aug 08. Download from: https://www.omgwiki.org/MBSE/lib/exe/fetch.php?media=mbse:patterns:incose_north_texas_pgm_08.08.2023_v1.2.2.pdf.
- ———. 2024. "All Decisions are Reconciliations of Inconsistencies: Preparing for the Digital Thread and Machine Learning." INCOSE 2024 International Symposium, Dublin, IE.
- SEI. 2010. CMMI® for Development, Version 1.3 CMMI-DEV, V1.3 CMMI Product Team Improving Processes for Developing Better Products and Services. Technical Report
- CMU/SEI-2010-TR-033 ESC-TR-2010-033, Carnegie Mellon University Software Engineering Institute.
- Shannon, C. 1948. "A Mathematical Theory of Communication." *Bell System Technical Journal* 27 (3): 379–423.
- Shibberu, Y. 1994. "Time-Discretization of Hamiltonian Dynamical Systems." *Computers Math. Applic.* 28 (10-12): 123-145.
- Toth, P., D. Rezende, A, Jaegle, S. Racanière, A. Botev, and I. Higgins. 2020. "Hamiltonian Generative Networks." 2020 International Conference on Learning Representations, Addis Ababa, ET.
- Trees, L., M. McCulloch, and N. Witt. 2021. Knowledge Management Trends: Survey Report. American Productivity & Quality Center, Houston, US-TX.
- Valerdi, R., B. Boehm, and D. Reifer. 2003. "COSYSMO: A Constructive Systems Engineering Cost Model Coming of Age." INCOSE 13th Annual International Symposium, Crystal City, US-VA.
- Walden, D., et al., eds. 2023. *Systems Engineering Handbook: A Guide for System Life Cycle Processes and Activities*, Fifth Edition. International Council on Systems Engineering, San Diego, US-CA: Wiley.

ABOUT THE AUTHOR

> continued on page 26

Realizing the Promise of Digital Engineering: Planning, Implementing, and Evolving the Ecosystem

William D. Schindel, schindel@ictt.com

■ **ABSTRACT**
Gaining benefits of digital engineering is not only about implementing digital technologies. An ecosystem for innovation is a system of systems in its own right, only partly engineered, subject to risks and challenges of evolving socio-technical systems. This paper summarizes an aid to planning, analyzing, implementing, and improving innovation ecosystems. Represented as a configurable model-based reference pattern used by collaborating INCOSE working groups, it was initially applied in targeted INCOSE case studies, and subsequently elaborated and applied to diverse commercial and defense ecosystems. Explicating the recurrent theme of consistency management underlying all historical engineering, it is revealing of digital engineering's special promise, and enhances understanding of historical as well as future engineering and life cycle management. It includes preparation of human and technical resources to effectively consume and exploit digital information assets, not just create them, capability enhancements over incremental release trains, and evolutionary steering using feedback and group learning.

■ **KEYWORDS:** digital ecosystem; digital engineering; digital thread; digital twin; collaboration; MBSE

INTRODUCTION

Many large-scale human endeavors have grown up and proliferated through the evolutionary forces of large-scale interactions and selection processes; however, as interacting systems of systems, they have not been consciously human engineered in the traditional sense. Human-performed systems of innovation include interacting elements such as competitive markets, scientific research, engineering, production, distribution, sustainment, and regulatory processes, and other life cycle management familiar to the systems engineering community (ISO 2015, INCOSE 2015). In the natural world, systems of innovation provide a much longer history for discovery and study than the more recent human-performed cases (Schindel 2013). For this paper's interest in human-performed cases for human use, we define "innovation" as delivery of significantly increased stakeholder value (Schindel, Peffers, et al. 2011).

The term "ecosystem," borrowed from the life sciences, has become more frequently applied to label the human-performed case, out of recognition of the vast extent, complexity, and dynamic evolution of the human-performed cases. Systems engineers less familiar with model-based systems engineering (MBSE) details are encouraged to view this approach as a systems view of that ecosystem and systemic impacts of information, not the details of models. The descriptive backbone of this article is the formal INCOSE Innovation Ecosystem Reference Model, configurable across diverse specific cases. (Since this paper is about that formal reference model, terms which are modeled class names from that reference model are shown in title case as they appear in the named model components.)

The engineering community is certainly not without high value historical models of at least portions of the human-performed Innovation Ecosystem. The above-referenced ISO standard and INCOSE Handbook, the ubiquitous "Vee" model, US Department of Defense (DoD) and enterprise-specific models, new model-based standard efforts to describe the Model-Based Enterprise, and others provide vital guidance. Out of respect for those historical assets and the importance of building upon them, they are accommodated within and mate up

with the larger-scale Innovation Ecosystem reference model's configurations referenced in this article.

Why is an ecosystem-level model needed? Smaller scale models have served to inform teams about what work needs to be done, coordinate flows of information, plan information systems, and other purposes. Is there really a need for an ecosystem level reference? Do our innovation ecosystems work well enough, and do we understand them well enough? Consider the following.

Ecosystem-level efforts and issues are arising that challenge our group-level abilities to effectively understand (individually and together) and communicate about the innovation ecosystem across life cycles, and particularly so while that ecosystem itself is evolving and the stakes are rising. We are increasingly interested in how to understand the basis of performance of the ecosystem as a whole (as in its timely delivery of competitive solutions) through its system components and their organization—for performance improvement, robustness, pathology, and security reasons. How do we integrate across supply chains? Are there other effective architectures besides historical original equipment manufacturer (OEM) and captive supplier relationships? How can we improve the real effectiveness of those or other combinations? Can we even effectively communicate about this subject without a shared neutral reference model? What is the connection of the engineering community's interest with the business management community's interest in "business ecosystems" (Jacobides 2017)?

Growth in conversations about "digital engineering," "digital twins" and "digital threads," all illustrate a growing need for foundational insight to support the "buzz" and to better connect to history even where departures are needed. The Innovation Ecosystem Reference Model described in this paper focuses on such a set of ecosystem issues. Following a brief introduction to the structure of the reference model, this article summarizes selected aspects which related experience has shown provide important insight and understanding worthy of increased attention:

1. Ecosystem-level capabilities' connection to underlying interactions;
2. Connecting historically understood business processes to evolving digital infrastructure;
3. Consistency Management's connection to realizing the promise of digital engineering;
4. Effectiveness of distributed, multi-level group learning across an ecosystem;
5. Group trust in the credibility of models;
6. Managing the proliferation of virtual model diversity and instances;
7. Effective evolution of the ecosystem itself—including implementation challenges.

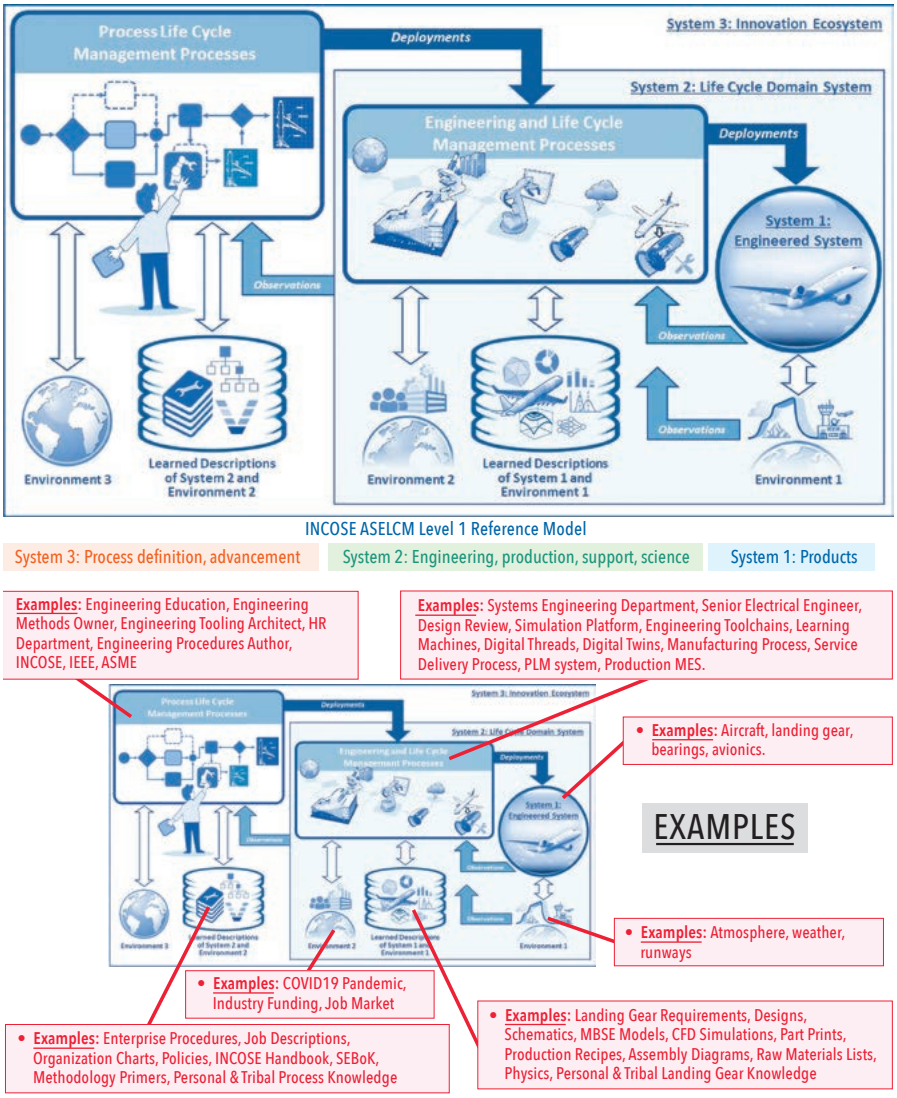


Figure 1. Level 0 Logical Architecture, Systems 1, 2, and 3

SELECTED ASPECTS OF THE INNOVATION ECOSYSTEM PATTERN

The reference model was proposed in a series of papers to describe adaptive purpose-seeking innovation ecosystems (Beihoff and Schindel 2011 and Schindel 2013). It was then elaborated during a multi-year INCOSE joint project of the Agile Systems Engineering and MBSE Patterns Working Groups to study agility across a range of aerospace and defense programs by leading enterprises (Schindel and Dove 2016; Dove, Schindel, and Scrapper 2016; Dove and Schindel 2017; Dove, Schindel, and Hartney 2017; Dove, Schindel, and Garlington 2018; and Dove and Schindel 2019). Since that time, it has been further elaborated by the MBSE Patterns Working Group to study issues listed in the introduction across other enterprises, and migrated into a generic configurable S*Pattern expressed in the Object Management Group (OMG) System Modeling Language (SysML®). At the time of this writing, it is also being applied as a reference model in joint publication projects by AIAA, INCOSE, and others to study a series of Digital Twin and Digital Thread cases and principles. This article summarizes aspects of the reference pattern translated from its more detailed OMG SysML version, using accurate but less formal graphic renditions, for ease of comprehension.

Reference Model Structure. Figures 1-3 informally summarize the formal model's logical architecture, Levels 0-2, the first three decomposition levels of the logical architecture.

By Level 2, these separate the roles played by ecosystem information classes from the business and technical processes that produce and consume that information. The blocks shown represent generic

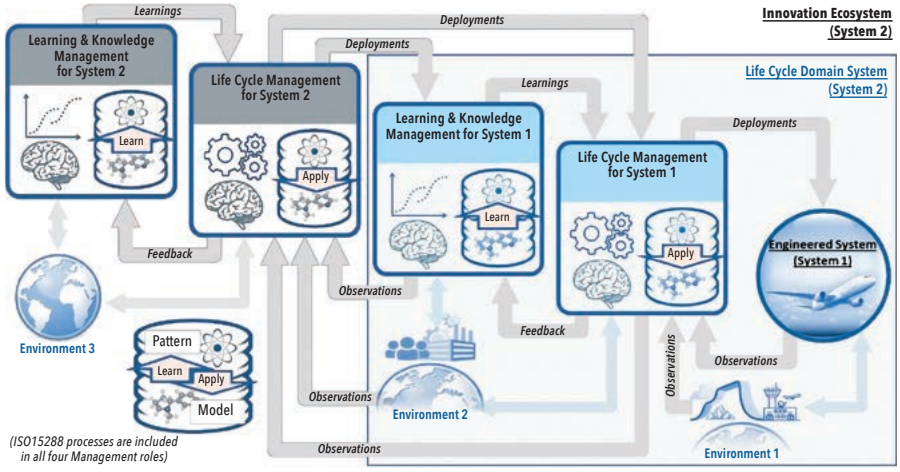


Figure 2. Level 1 Logical Architecture–separates learning from applying what is learned

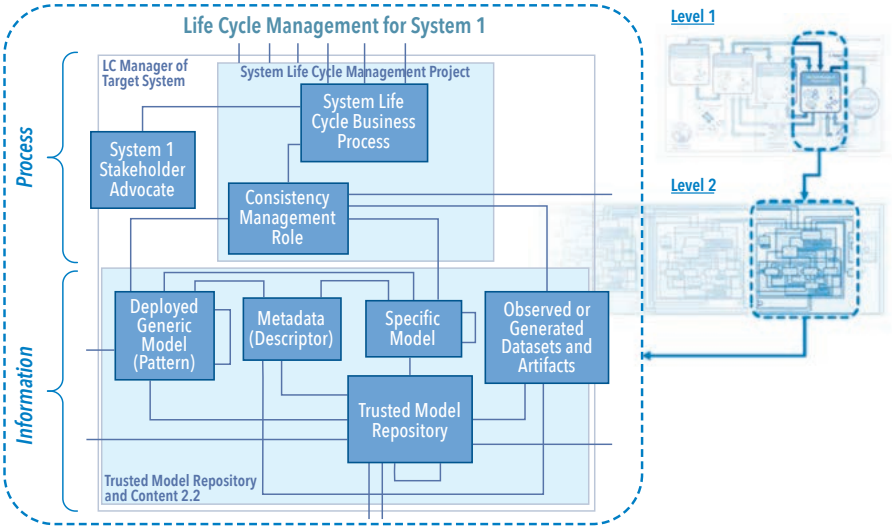


Figure 3. Level 2 Logical Architecture—Process Roles versus Information Roles

configurable logical roles (behaviors), not specific methods, until they are configured. Prominent in this decomposition are three reference boundaries, for defined Systems 1, 2, and 3:

System 1—The Engineered System of Interest: Viewed at any and all times in its life cycle.

System 2—The Life Cycle Domain System: The environment with which the Engineered System interacts, across its life cycle. This includes all Life Cycle Management systems responsible for the Engineered System (research, engineering, manufacturing, distribution, markets, operations, sustainment). System 2 is responsible to observe and learn about System 1 and its environment, not just engineer and deploy it. A model or artifact describing System 1 is a subsystem of System 2, which also includes collaborating users of that information.

System 3—The Innovation Ecosystem: Includes the system responsible to plan, deploy, and evolve System 2, responsible to observe and learn about System 2 and its environment. Writing and reading this article are System 3 activities, as are many other technical society activities intended to improve the future System 2's of the world.

As an MBSE S*Pattern (a reusable, configurable MBSE model), the reference model has more components than just logical architecture, including stakeholder features (Figure 4) describing configurable ecosystem capabilities, functional interactions between functional roles, interfaces and systems of access, allocations to design components, attributes, and other components, mapped into OMG SysML. The details of the pattern methods of representation are beyond the scope of this ecosystem model article but described further in Schindel and Peterson (2016), INCOSE Patterns WG (2019b), and Patterns WG (2020a).

1. **Ecosystem-level capabilities' connection to underlying interactions.** Our first concern for an Innovation Ecosystem is for its capabilities. Figure 4 summarizes the modeled Stakeholder Features built into the configurable reference model. For a given current or planned ecosystem of interest, these are configured by variably populating them (multiple instances in some cases) or not, and setting their attribute values, similarly, to viewing the Innovation Ecosystem as a configurable Product Line Engineering model—but as a product line of configurable ecosystems. The resulting configured feature model represents the overall capabilities of an innovation ecosystem of interest—whether past, current, or future, whether favorable or unfavorable, for analysis, planning, communication, or other purposes. A series of these configurations represents a planned or real trajectory of ecosystem capabilities evolution over time. Figure 4 shows sample capabilities (features and their attributes) from ISO15288 systems engineering, along with agile engineering capabilities, digital threads and twins, and other capabilities at a stakeholder level. The feature attributes (properties) shown include Feature Primary Key attributes whose configured values invoke modeled population of specific ecosystem interactions of the roles from Figure 3, providing technical behaviors delivering the configured capabilities.

2. **Connecting historically understood business processes to evolving digital infrastructure.** The System Life Cycle Business Processes shown in the upper sections of Figures 3 and 5 represent either traditional or evolving business processes from the ecosystem (supply chain partners, enterprises, etc.) description of existing or planned business processes for research, engineering, production, distribution, sustainment, and other life cycle management processes. It is these processes (typically some targeted subset of them) that the Digital Engineering enhancements shown in other blocks are to advance, as discussed in the following sections. The important point here is that the advanced digital engineering roles to be discussed next are by this means connected to the more familiar existing, traditional, or planned local reference business process framework they are to serve and enhance. We are now ready to connect those business processes to the digital engineering promise, using the key insight of the Consistency Management role introduced in Figure 3.

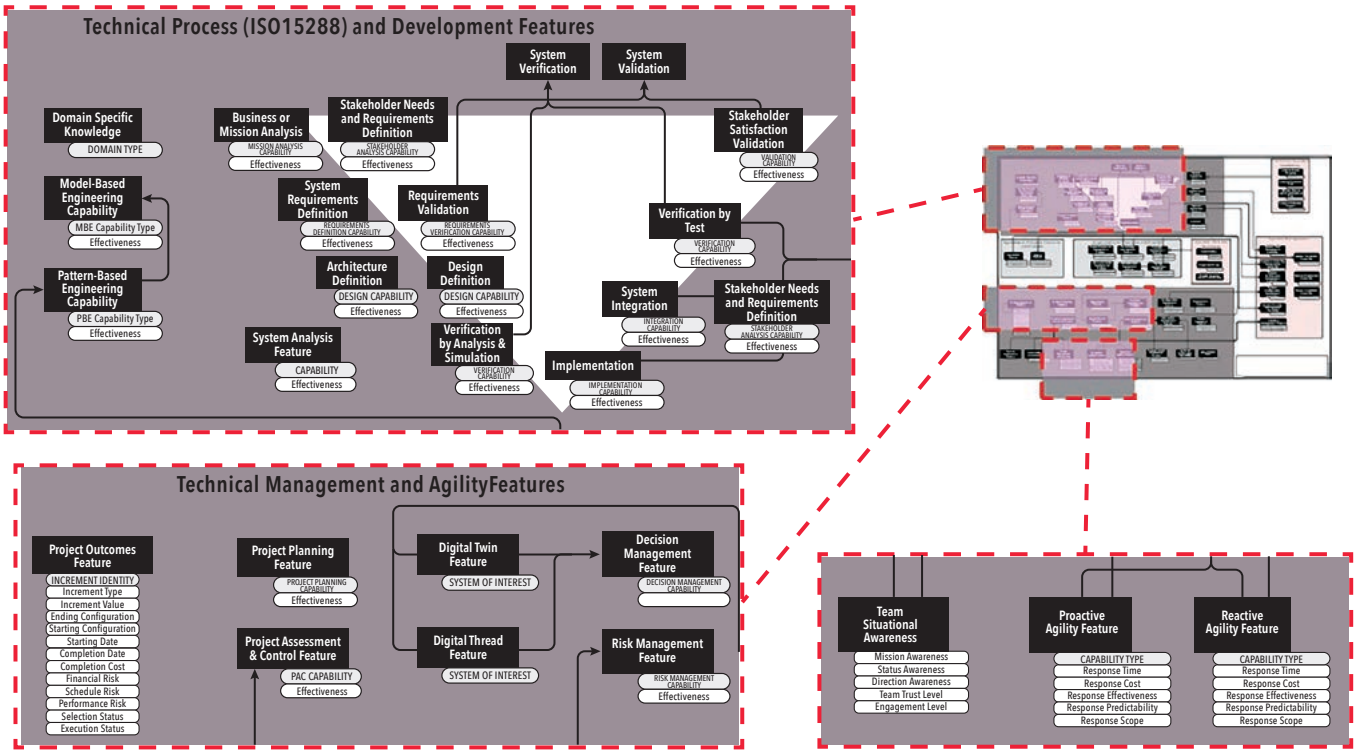
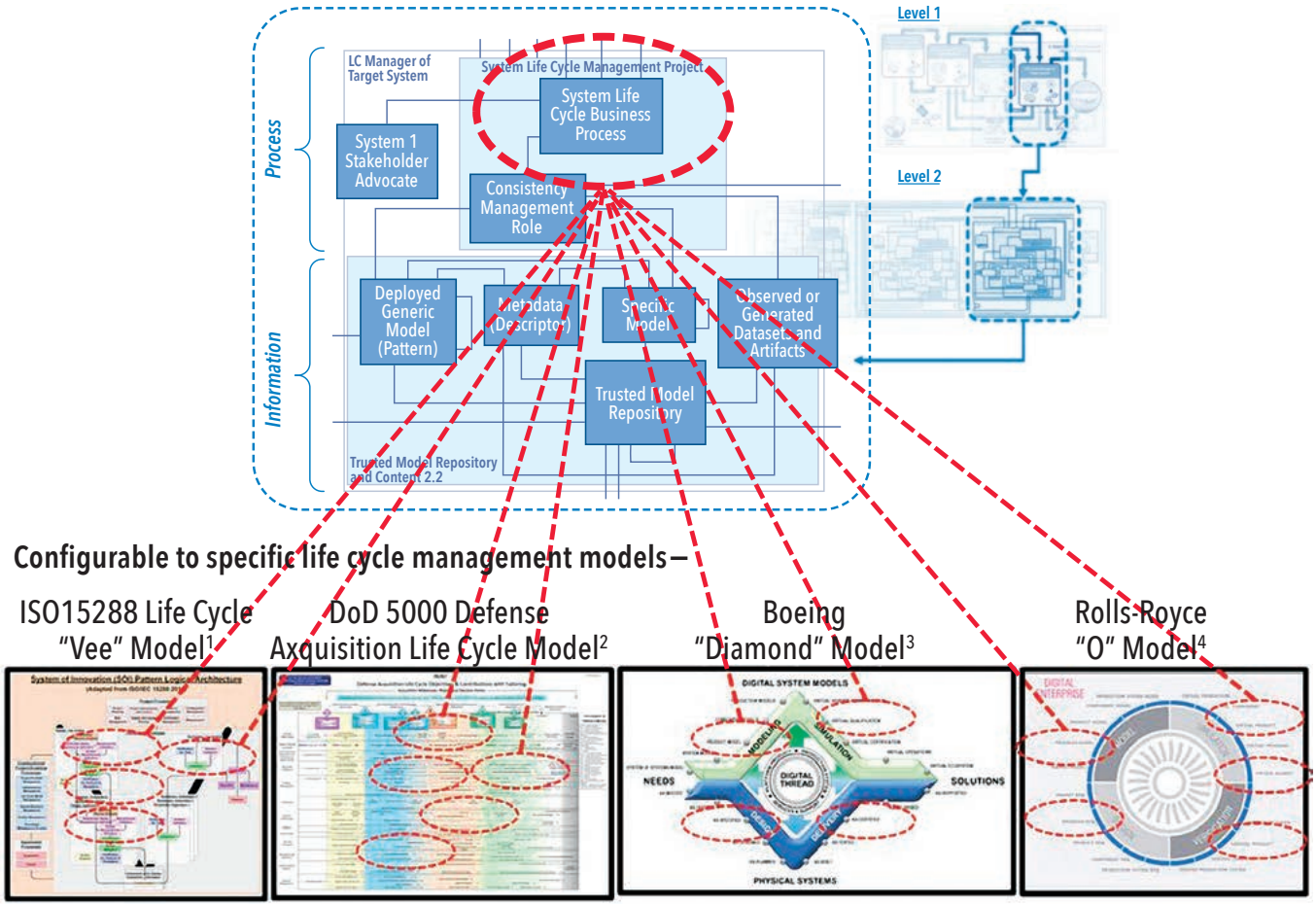


Figure 4. Configurable stakeholder features (Innovation Ecosystem System 2 capabilities)



Excerpted or adapted from: (1) ISO15288 and INCOSE SE Handbook; (2) DoD5000 Wall Chart; (3) AIAA Sci Tech, 01.2020, J. Matakeyama; (4) AIAA DEIC Digital Twin Subcommittee, 04.08.19 Donaldson, Flay, French, Matlik, Myer, Pond, Randjielovic

Figure 5. Business processes of the ecosystem appear in the Configurable Reference Model

3. **Consistency Management's connection to realizing the promise of digital engineering.** The traditional systems engineering "Vee diagram" in the lower left of Figure 5, along with the other adjacent US DoD and enterprise models, all remind us that all engineering methods in one way or another inherently manage a series of "gaps" into acceptable "consistencies:"
   - Consistency of formally recorded system requirements with stakeholder needs
   - Consistency of system designs with system requirements
   - Consistency of virtual simulations with empirical measurements (model verification, validation, and uncertainty quantification VVUQ)
   - Consistency of system component production with system design
   - Consistency of system performance with system requirements
   - Consistency of system operation with system requirements and design
   - Consistency of system sustainment with system requirements and design
   - Consistencies of many aspects with applicable technical standards, regulation, and law
   - Consistencies of many aspects with learned experiences, formal patterns of requirements and design, physical science, product line rules, architectural frameworks, shared ontologies, domain specific languages, and model semantics
   - Managed consistencies of the Digital Thread and Digital Twin
   - Many other types of consistencies.

Nearly all of these were also required consistencies in the traditional, more "tolerant" human-performed ecosystems lacking as much digital technology, even if not recognized as so.

The Consistency Management Role in Figure 3 represents the configurable set of process roles responsible for consistency management—whether performed by humans or automated, and whether performed well or not. It is understandable that much of this role has historically been performed by humans, because of required skills, judgement, experience, and information forms.

The digital engineering and modeling community finds itself in frequent conversations about a perceived need for a "single source of truth" or "authority", reflecting frustrations with diverse and inconsistent information about systems. Figure 6 reminds us this situation is not as simple as might be assumed, showing the three main sources of information in any ecosystem:

T1. What the stakeholders say (market and sponsor truths);

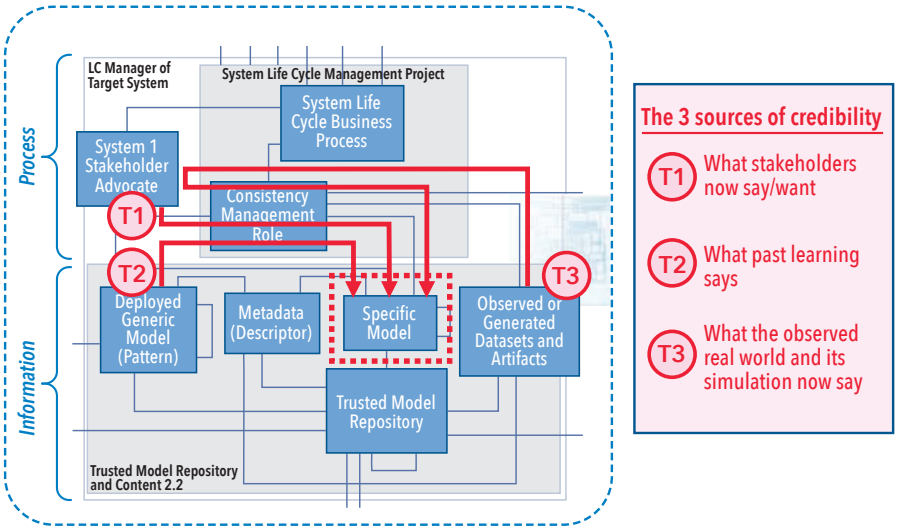


Figure 6. Roots of the consistency management challenge

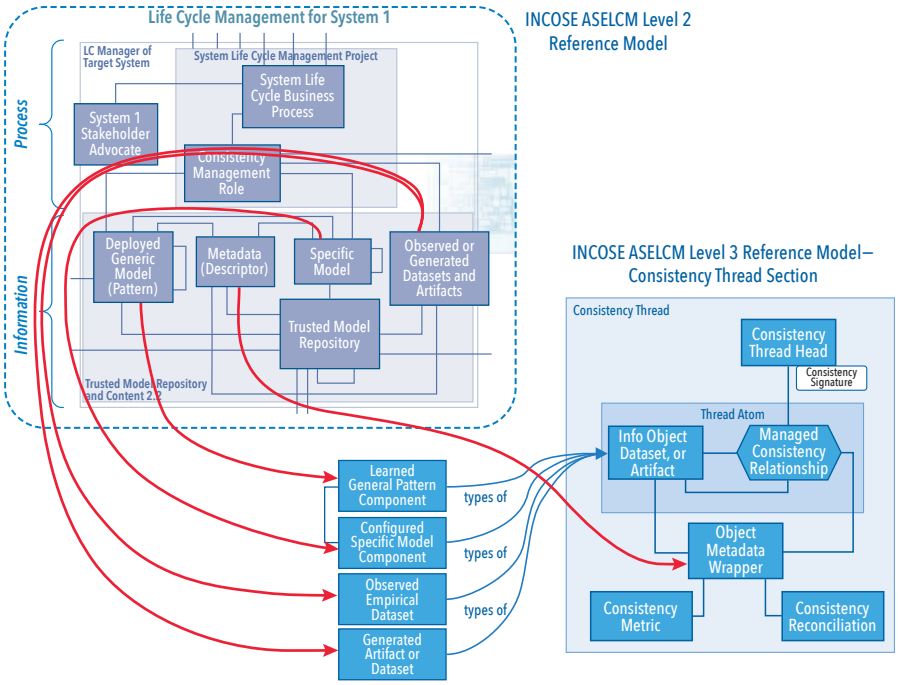


Figure 7. The Consistency Thread—Antecedent of the Digital Thread

T2. What experience says (accumulated, hard-won past discoveries; includes physical science);
T3. What empirical observation says (observation, measurement, experiment).

The challenge is that these three sources will frequently be inconsistent (disagree with each other). The Figure 3 Consistency Management Roles of engineering and other life cycle management processes historically must recognize (detect) those inconsistencies and reconcile them. While the resulting reconciliations may be considered "authoritative" or "single", they are short-lived.

The rise of interest in digital thread and digital twin methods fits into this consistency management perspective. This is currently being applied in a series of industry case studies by AIAA with INCOSE support. In the case of the digital twin, it reminds us of the importance of (1) managing both consistency between the virtual simulation model and the real system it simulates, and (2) managing the consistency of business processes and their information with what the trusted digital twin virtual model tells them. In the case of the digital thread (Figure 7), the central issue of the "thread" is managed consistency between a range of information objects along that thread. (Even sources external

to the thread generate information samples within it.) Historical predecessors to the digital thread bring important perspective to this evolution. Depending on industry domain, these include SAE (2016), AIAG (2006), and ISO (2016).

Because consistency gaps are often rooted in conflicting interests of different parties, the Consistency Management role is the potential site for impactful multi-party collaboration across the ecosystem or supply chain. Enabling this collaboration with explicit models of the respective parties' collaboration configuration spaces makes it easier to understand it as a problem of differential or modular games (Schindel and Seidman 2021, Schindel 2021, and Leitmann 1975).

The history of consistency management across the product life cycle has seen varied gap sizes at some stages versus others. This has meant that production, logistics, sustainment, and operation consistency gaps may be larger or longer-lived until reconciled. The ASELCM analysis framework helps us to see that these may be viewed not just as consistency gaps in System 1's life cycle (as viewed by System 2), but also as consistency gaps in the description of System 2 (as observed and modeled by System 3). This suggests another way to recognize and head off these gaps sooner and at less cost.

Many benefits sought through transformation to Digital Engineering have been discussed widely, such as basic issues of improved information accessibility, early virtual verification through simulation, and other gains. The Innovation Ecosystem Pattern reminds us, through the Consistency Management Role, of the wider promise that a variety of Consistency Management issues at the heart of every life cycle stage may ultimately be attacked more effectively through the aid of digital information technologies that assist in Consistency Management. These include semantic web technologies, machine learning, consistency thread signatures, configurable patterns, and pattern-based model metadata (Herzig and Paredis 2014; Herzig, Qamar and Paredis 2014; Kerstetter and Woodham 2014; Redman 2014; and Patterns WG 2020b).

4. **Effectiveness of distributed, multi-level, group learning across an ecosystem.** The promise of digital engineering should not be to optimize single program outcomes while "forgetting" what is learned when the next program starts, nor to arbitrarily isolate one team's learning from other teams within a shared community. Traditional descriptions of the systems engineering life cycle processes (for example, ISO 15288, INCOSE Systems Engineering Handbook, etc.) describe all the processes a program should follow to generate all the information needed across the life cycle, but are relatively silent on the questions: "What about what we already know?" and "What about the impact on future programs of what we learned the hard way on past programs?" This begs the question of what is really meant by "what we learned" and "what we know"—what is group knowledge?

The management of balancing acquisition and validation of new information versus exploiting existing information is also frequently omitted in those descriptions, left for separate consideration. This is in ironic contrast to one of the great successes of modern signal processing and control theory—the optimal mixing of past experience with new information, in the presence of uncertainty, discussed further in a later section below.

The Innovation Ecosystem model views effective learning as not just accumulation of information as IP assets, but instead as improvement of future performance across the ecosystem based on past experience. This especially includes more effective application of learned results that were acquired by different people at different times. The Ecosystem Pattern makes explicit the two roles of learning and subsequent application, and their integration—refer to Figure 2. That integration can include starting new program executions by configuring general learned System 1 and 2 patterns in a form specific to the new program. To the degree it is performed, this capability is referred to by the MBSE Patterns Working Group as pattern-based systems engineering (PBSE) (Patterns WG 2020a).

Key System 2 capabilities that, if present, contribute to that performance include:

Synthesizing Generalization: Distillation of learning as model-based abstractions, curated at the abstraction hierarchy level where they can have the greatest future impact. The "up" (Learn) arrows in Figure 2;
Validation for Context of Use: Reusable configurable model verification, validation, and uncertainty quantification, credibility assessment, establishment of pattern metadata on provenance, credibility, and intended range of use. More on this in the next section;
Configuring Specialization: Harvesting of accumulated learned patterns at the place and time (and for the people where) they are impactful, through their configuration into new projects as part of the initiation of those projects. The "down" (Apply) arrows in Figure 2.

After it is understood that configuration space is not "flat", but organized by evolving patterns at different abstraction levels, two challenging opportunities can be better understood:

The dynamic evolutionary nature of semantic interoperability: Domain-specific ontologies will continue to spring up as long as new system interactions and interaction levels are pursued describing new phenomena—and this is forever. One of the competencies required of the digital ecosystem is continuous collaborative synthesis of new, often higher-level, semantic frameworks for interoperability (Schindel 2020).
The opportunities for sharing and ownership at different levels: Shared frameworks across large ecosystems can lift the fortunes of all boats, as in the case of pre-competitive standards shared by competitors—but can be perceived as counter to the interests of individual suppliers, customers or employees who wish to own, control, or be differentiated by less shared models. Non-flat pattern hierarchy allows for mixing of shared ecosystem-wide generic patterns with compatible specializations controlled or licensed by competitive ecosystem members, providing simultaneous differentiation and compatibility.

Digital Engineering offers special promise in the above areas through the use of information technologies that empower virtual models, their generalization and configuration, and related processes with capacity exceeding human performance alone. But it also demands new human skills and orchestration on the human side of the Digital Engineering partnership. Model-based group learning is also related to issues of trust in model credibility, discussed next.

5. **Group trust in the credibility of models.** Model credibility involves the verification and validation of a model's fitness for use for a stated purpose (ASME 2018), explicit tracking of related uncertainties (NAE 2012), and larger issues of propagation of trust (Rhodes 2018). The growing proliferation of model instances, types, and uses means that more uniform model metadata approaches are becoming important to describe those diverse assets in more uniform ways—somewhat like the emergence of bar code labels on supermarket products. Because there are a variety of model credibility factors that may be applied, credibility assessment frameworks. (CAFs) can serve

a useful purpose as part of that model metadata (Kaizer 2018). The INCOSE MBSE Patterns Working Group has developed a model characterization pattern (MCP) descriptive of models of all types (Patterns WG 2019a), building in enterprise-configured CAFs.

Many aspects of the engineering cycle are concerned with determining whether aspects of related information are worthy of trust for use in a given context. When this interest is translated to operate with virtual models, it is bolstered by the powerful technical toolset developed over the longer history of the (model-based) scientific revolution, in which the credibility of candidate models, and their repeated uses across different instances are both central. Computational model verification, validation, and uncertainty quantification (VVUQ) is a vital portion of this infrastructure.

Group trust in model credibility is not just a technical matter of the fidelity of the models themselves. Group trust is a socially transmitted property, in which additional credibility factors such as trust in intermediate messengers and interpreters carries great weight (Rhodes 2018). Models of how credibility (or doubts of credibility) are propagated through ecosystems can illustrate the contest of multiple factors impacting group trust, distrust, confidence, or doubt. The above credibility assessment frameworks (CAFs) preserve for future reference the basis on which credibility was assessed for a given model, whether it later proves to be valid or not.

6. **Managing the proliferation of model diversity and instances.** Such model credibility information is a special case of larger class of model metadata—information outside a virtual model that describes the virtual model. Model metadata can variously include description of a model's focal subject, structure, algorithms, intended model use and context of that use, model provenance, model credibility, the nature and scope of the virtual model, and refer to related model artifacts, datasets, and life cycle maintenance history. Figure 3 graphically notes the role that model metadata plays within the innovation ecosystem, describing diverse virtual models (and datasets) to their potential users, as a kind of uniform "labeling wrapper" of evolving virtual models. While it has been common to consider many aspects of information technology in planning Digital Engineering, awareness of the broader roles of virtual model metadata deserves expanded awareness.

The diversity of types of virtual models includes computational models (simulations of all kinds) and descriptive MBSE models but can also other forms of formalized standards-based data structures. Simulations alone may include physics-based finite element analysis (FEA) and computational fluid dynamics (CFD) discretized continuum simulations, ordinary differential equation-based simulations, machine learning models and other forms of data-driven models, and others. Adding to this diversity are varied model author styles, computing environments, and methodologies for model verification, validation, uncertainty quantification, and credibility management. The resulting explosion of model diversity as well as model quantities is exacerbated by increasing separation between model authors and model users.

The Model Wrapper generic metadata role shown in Figure 3 serves purposes similar to the package labeling, inserts, and supplemental downloads common to consumer products. Imagine walking into a modern supermarket, big box store, or distributor web site, and finding that all the package and shelf labeling and explanations have disappeared except for the ability to directly view the products (remember earlier open-air market bazaars). This conveys some idea of the current situation concerning proliferation of thousands of models within an enterprise, and even more pronounced across a future multi-enterprise ecosystem in which exchange of models occurs. Generic metadata frameworks for engineering models, such as the model characterization pattern (Patterns WG 2019a) and model identity card (MIC) (Goknur 2015) are key enablers to the effectiveness of the digital engineering in the Innovation Ecosystem.

7. **Effective evolution of the ecosystem itself—including implementation.** Among the promises of the digital engineering ecosystem are its own adaptability, as future environments and market situations demand. An essential capability described by the Innovation Ecosystem Pattern is that adaptability. In Figures 1 and 2, System 3 is concerned with adaptability of System 2, beginning by observing and representing it, followed by analyzing and deploying adaptations to System 2 instances. Viewing System 2 through the lens of systems engineering, this includes implementation. The deployed or updated "design components" of System 2 are collaborating people, enterprises, information systems, equipment, and facilities of System 2, and how they are organized (interact with each other), planned over agile release train configurations of the System 2 pattern. In addition to the challenges of engineering, such adaptation implementation also carries all the challenges of enterprise organizational change management (OCM) (Kotter 2014). Just as the forces of multi-stage selection operate over the life cycles of the engineered products of System 1, (other) multi-stage selection forces also shape the evolution of System 2 (Patterns WG 2020a). Understanding those forces is essential to the conscious design of (or at least influence on) the evolution of System 2. For complex business ecosystems involving multiple partners, not only is the alignment of their technical capabilities vital, but also the alignment of their business interests and incentives. These issues should remind us that successful collaboration across System 2 requires more than just a digital medium for that collaboration. Heeding the wisdom of the lengthy related literature (for example, Kotter 2014) on organizational change is a key part of implementation planning.

In the language of business management community, "business ecosystem" has come to refer to particular ecosystem architectures (for System 2) which operate flexibly as small "markets" in which modularity of the System 1 technical approach encourages a more dynamic (and accordingly less stable) arrival and departure of competing candidate System 2 partners offering contributions to solutions. (Jacobides 2017)—this in contrast to traditional OEM plus captive smaller suppliers linear supply chain network models. Both the advantages and disadvantages of such approaches can be seen in the real history of the personal computer (PC). Early PCs were proprietary closed architectures from competing end product suppliers. This picture was disrupted when IBM opened the digital product's electronic circuit card bus specification and business ecosystem to third party suppliers who could directly supply add-in circuit cards to the end user. The market dramatically expanded through innovative add-ons, lifting all boats, but eventually driving the originator (IBM) of that approach out of the market. These are not just stories of the System 1 architecture, but also of the System 2 architecture.

Selection processes performed by System 2 and 3 can be understood as cycles of their Consistency Management Roles (see Figure 3), selecting opportunities, requirements, candidate designs, and other aspec ts of both System 1 products and System 2 enterprise designs. In those cycles, Digital

Engineering offers special promise for exploiting the following "Goldilocks" insight from the successful history of engineering certain challenging systems:

More consideration of empirical inputs: When more agility was needed to converge sooner on the real needs of stakeholders and real solutions to them, the pioneers of agile engineering introduced cycles that paid earlier, more frequent, and ongoing attention to incoming reality signals from System 2 experiment and empirical measurements involving real world signals instead of isolated planning. The upside of this produces early minimum viable products (MVPs), rapid learning by individuals and small teams, and successful "pivots". On the downside, it may miss exploitation of what was already discovered and can produce ill-conceived course changes chasing noisy data.

More consideration of patterns of experience: When more instances of variant products proliferated to address different market segments, the pioneers of design patterns and product line engineering introduced cycles that paid more attention to shared historical patterns of product designs, requirements, and other common but configurable assets. The upside of this produces increased IP leverage and flexibility. If overperformed, it risks constraints that may miss external shifts and trends, dragging along too much of the past.

Goldilocks as Kalman: More optimal mixing observation and experience: Formal systems engineering process descriptions often tell us all the things we should do to learn what is needed for good life cycles but may be silent on the questions "what about what we already know?", and "how can we discover new things sooner?", addressed by the two complementary points above. In one of the most impactful examples of breakthrough engineering through applied mathematics, Rudolf Kalman introduced an approach to optimal mixing of these two in the presence of uncertainty, the Kalman Filter approach to Bayesian estimation, power navigation to landing on the Moon, world-wide personal communication systems, countless industrial control systems, and other applications of this combination. Digital Engineering offers a medium in which the Consistency Management Role of Figure 3 can be advanced to leverage those insights in support of human decision-making (Schindel 2017b). Improving ontological patterns and their use can improve meaning and understanding of empirical data from improved sensory and observational networks. Collaborative ecosystem efforts to create capabilities such as joint all domain command and control (JADC2) can benefit from these historical insights (CRS 2021).

CONCLUSIONS, NEXT STEPS, AND AN INVITATION

The seven selected aspects of the Ecosystem Pattern discussed in this paper demand greater community-wide attention in planning and analyzing digital ecosystems, and the neutral descriptive framework described offers a means of doing so. The systems engineering community has a shared interest in the network benefits of community-wide advancement of ecosystems for digital engineering. The INCOSE MBSE Patterns Working Group continues to pursue the discovery and expression of explicit model-based patterns, which fuel digital ecosystems as "water through their pipes", but which also represent those ecosystems themselves (Patterns WG 2021).

The Patterns Working Group conducts most of its activities as collaborations with other INCOSE and additional technical society groups, to advance awareness and the state of practice. Interested readers are invited to participate in this progress and learn along with us about use of the related aids and examples that this reference pattern supports:

- Details of the Ecosystem Pattern, now being tested in its OMG SysML form
- The Ecosystem Pattern as a digital engineering capability planning aid (Patterns WG 2020c)
- Basics of S*Models, S*Patterns, and the S*Metamodel (Patterns WG 2019b)
- Domain specific applications of model-based patterns (Patterns WG 2021) ■

ACKNOWLEDGEMENTS

The Ecosystem Pattern is informed by the practices and ideas from numerous pioneers and practitioners. The encouragement, suggestions, and inspiration from Rick Dove, chair of the INCOSE Agile Systems Engineering Working Group, the lead team of the INCOSE Agile Systems Discovery Project, and the membership of INCOSE MBSE Patterns Working Group, along with the anonymous reviewers of this paper, are all acknowledged with gratitude.

REFERENCES

- AIAG. 2006. "APQP & PPAP Requirements for Automotive." Automotive Industry Action Group, Southfield, US-MI. https://www.techstreet.com/standards/aiag-ppap-4?product_id=1257705.
- ASME. 2018. "VV40: Assessing Credibility of Computational Modeling through Verification and Validation: Application to Medical Devices." New York, US-NY. https://www.asme.org/codes-standards/find-codes-standards/v-v-40-assessing-credibility-computational-modeling-verification-validation-application-medical-devices.
- Beihoff, B., and W. Schindel. 2011. "Systems of Innovation I: Summary Models of SOI Health and Pathologies." INCOSE 2011 International Symposium, Denver, US-CO.
- CRS. 2021. "Joint All-Domain Command and Control (JADC2): Background and Issues for Congress." Congressional Research Service, Washington, US-DC. https://crsreports.congress.gov/product/pdf/R/R46725/2.
- Dove, R., and W. Schindel. 2019. "Agile Systems Engineering Life Cycle Model for Mixed Discipline Engineering." INCOSE 2019 International Symposium, Orlando, US-FL.
- Dove, R. and W. Schindel. 2017. "Case study: Agile SE Process for Centralized SoS Sustainment at Northrop Grumman." INCOSE 2017 International Symposium, Adelaide, AU. https://www.omgwiki.org/MBSE/lib/exe/fetch.php?media=mbse:patterns:is2017--northrup_grumman_case_study_dove_and_schindel_bp.pdf.
- Dove, R., W. Schindel, and K. Garlington. 2018. "Case Study: Agile Systems Engineering at Lockheed Martin Aeronautics Integrated Fighter Group." INCOSE 2018 International Symposium, Washington, US-DC. https://www.omgwiki.org/MBSE/lib/exe/fetch.php?media=mbse:patterns:is2018_-_aselcm_lmc_case_study.pdf.
- Dove, R., W. Schindel, and W. Hartney. 2017. "Case Study: Agile Hardware/Firmware/Software Product Line Engineering at Rockwell Collins." IEEE 11th Annual International Systems Conference. Institute of Electrical and Electronic Engineers, New York, US-NY. https://www.omgwiki.org/MBSE/lib/exe/fetch.php?media=mbse:patterns:pap170424syscon-casestudyrc.pdf.
- Dove, R., W. Schindel, and C. Scrapper. 2016. "Agile Systems Engineering Process Features Collective Culture, Consciousness, and Conscience at SSC Pacific Unmanned

Systems Group." INCOSE 2016 International Symposium, Edinburgh, GB-SCT. https://www.omgwiki.org/MBSE/lib/exe/fetch.php?media=mbse:patterns:is2016_—_autonomous_vehicle_development_navy_spawar.pdf.

- Göknur, S., C. Paredis, B. Yannou, E. Coatanéa, and E. Landel. 2015. "A Model Identity Card to Support Engineering Analysis Model (EAM) Development Process in a Collaborative Multidisciplinary Design Environment/" *IEEE Systems Journal*, IEEE, New York, US-NY. https://hal.archives-ouvertes.fr/hal-01184938/document.
- Herzig, S., and C. Paredis. 2014. "A Conceptual Basis for Inconsistency Management in Model-Based Systems Engineering." CIRP 2014 Design Conference, International Academy for Production Engineering, Paris, FR. https://www.sciencedirect.com/science/article/pii/S2212827114007586/pdf?md5=c9bdd8aba94e820ec43b56330225daa6&pid=1-s2.0-S2212827114007586-main.pdf.
- Herzig, S., A. Qamar, and C. Paredis. 2014. "Inconsistency Management in Model-Based Systems Engineering." 2014 Global Product Data Interoperability Summit, Southfield, US-MI. http://gpdisonline.com/wp-content/uploads/past-presentations/AC45_GeorgiaTech-SebastianHerzig-InconsistencyManagementInMBSE.pdf.
- ISO. 2015. "ISO 15288:2015 Systems and Software Engineering — System Life Cycle Processes." International Standards Organization, Geneva, CH. https://www.iso.org/standard/63711.html.
- ISO. 2016. "ISO 13485:2016 Medical devices — Quality Management Systems — Requirements for Regulatory Purposes." International Standards Organization, Geneva, CH. https://www.iso.org/standard/59752.html.
- Jacobides, M. 2017. "Towards a Theory of Ecosystems (with Phenomenological Preamble)." Keynote presentation at 5th International Conference of the Armand Peugeot Chair Paris, FR. https://chairgovreg.fondation-dauphine.fr/sites/chairgovreg.fondation-dauphine.fr/files/attachments/JCG%20CAP%20Paris%202017%20presentation%20S.pdf.
- Kaizer, J. 2018. "Credibility Assessment Frameworks – Personal Views." ASME Symposium on Verification and Validation, American Society of Mechanical Engineering, New York, US-NY. https://cstools.asme.org/csconnect/FileUpload.cfm?View=yes&ID=54674.
- Kerstetter, M., and K. Woodham. 2014. "SAVI Behavior Model Consistency Analysis." 2014 Global Product Data Interoperability Summit, Southfield, US-MI. http://gpdisonline.com/wp-content/uploads/past-presentations/AVSI-Kerstetter-SAVIBehaviorModelConsistencyAnalysis-CAE-Open.pdf.
- Kotter, J. 2014. *Accelerate: Building Strategic Agility for a Faster-Moving World*. Cambridge, US-MA: Harvard Business Review Press.
- Leitmann G. 1975. "Cooperative and Non-Cooperative Differential Games." G. Leitmann and A. Marzollo, editors. *Multicriteria Decision Making*. International Centre for Mechanical Sciences (Courses and Lectures) 211: Springer, Vienna, AT. https://doi.org/10.1007/978-3-7091-2438-3.
- NAE. 2012. "Assessing the Reliability of Complex Models: Mathematical and Statistical Foundations of Verification, Validation, and Uncertainty Quantification." National Academy of Engineering, Washington, US-DC. https://www.nap.edu/catalog/13395/assessing-the-reliability-of-complex-models-mathe matical-and-statistical-foundations.
- Patterns WG. 2021. INCOSE MBSE Patterns Working Group Web Site: https://www.omgwiki.org/MBSE/doku.php?id=mbse:patterns:patterns.
- Patterns WG. 2019a. "The Model Characterization Pattern: A Universal Characterization & Labeling S*Pattern for All Computational Models." V1.9.3. INCOSE Patterns Working Group web site, International Council on Systems Engineering, San Diego, US-CA. https://www.omgwiki.org/MBSE/lib/exe/fetch.php?media=mbse:patterns:model_characterization_pattern_mcp_v1.9.3.pdf.
- Patterns WG. 2019b. "Methodology Summary: Pattern-Based Systems Engineering (PBSE), Based On S*MBSE Models." INCOSE Patterns Working Group web site, International Council on Systems Engineering, San Diego, US-CA. https://www.omgwiki.org/MBSE/lib/exe/fetch.php?media=mbse:patterns:pbse_extension_of_mbse—methodology_summary_v1.6.1.pdf.
- Patterns WG. 2020a. "ASELCM Reference Pattern: Reference Configuration Stages for Models, Model Patterns, and the Real Systems they Represent." INCOSE Patterns Working Group web site, International Council on Systems Engineering, San Diego, US-CA. https://www.omgwiki.org/MBSE/lib/exe/fetch.php?media=mbse:patterns:configuration_stages_v1.4.5.pdf.
- Patterns WG. 2020b. "Consistency Management as an Integrating Paradigm for Digital Life Cycle Management with Learning." INCOSE Patterns Working Group web site, International Council on Systems Engineering, San Diego, US-CA. https://www.omgwiki.org/MBSE/lib/exe/fetch.php?media=mbse:patterns:aselcm_pattern_—_consistency_management_as_a_digital_life_cycle_management_paradigm_v1.2.2.pdf.
- Patterns WG. 2020c. "Example Use of ASELCM Pattern for Analyzing Current State, Describing Future State, and Constructing Incremental Release Roadmap to Future." INCOSE Patterns Working Group web site, International Council on Systems Engineering, San Diego, US-CA. https://www.omgwiki.org/MBSE/lib/exe/fetch.php?media=mbse:patterns:example_evolutionary_roadmap_v1.3.3a.pdf.
- Redman, D. 2014. "Importance of Consistency Checking in the SAVI Virtual Integration Process (VIP)." 2014 Global Product Data Interoperability Summit, Southfield, US-MI. http://gpdisonline.com/wp-content/uploads/past-presentations/SE_67_AVSI-Redman-ConsistencyCheckingInSAVI.pdf.
- Rhodes, D. 2018. "Interactive Model-Centric Systems Engineering (IMCSE)." Phase 5 Technical Report. SERC-2018-TR-104. Systems Engineering Research Center, Hoboken, US-NJ. https://apps.dtic.mil/sti/pdfs/AD1048003.pdf.
- SAE. 2016. "AS9145: APQP & PPAP Requirements for Aerospace and Defense." SAE International, Warrendale, US-PA. https://www.sae.org/standards/content/as9145/.
- Schindel, W. 2013. "Systems of Innovation II: The Emergence of Purpose." INCOSE 2013 International Symposium, Philadelphia, US-PA. https://www.omgwiki.org/MBSE/lib/exe/fetch.php?media=mbse:patterns:systems_of_innovation--the_emergence_of_purpose_v1.3.6.pdf.
- ———. 2021. "Variational Forces of Modularity: Coupled Macro and Micro Patterns in the Innovation Ecosystem." 2021 PLE Momentum Conference, Big Lever Inc, Austin, US-TX. https://www.omgwiki.org/MBSE/lib/exe/fetch.php?media=mbse:patterns:the_forces_of_modularity_v1.3.3.pdf.
- ———. 2017a. "MBSE Maturity Assessment: Related INCOSE & ASME Efforts, and ISO 15288." MBSE World Symposium, No Magic, Inc., Allen, US-TX. https://www.omgwiki.org/MBSE/lib/exe/fetch.php?media=mbse:patterns:model_based_maturitiy_planning_asme_incose_may_2017.pdf.

- ———. 2017b. "Innovation, Risk, Agility, and Learning, Viewed as Optimal Control & Estimation." INCOSE 2017 International Symposium, Adelaide, AU. https://www.omgwiki.org/MBSE/lib/exe/fetch.php?media=mbse:patterns:risk_and_agility_as_optimal_control_and_estimation_v1.7.2.pdf.
- ———. 2020. "SE Foundation Elements: Implications for Future SE Practice, Education, Research." INCOSE Vision 2035 Project, 21-22. International Council on Systems Engineering, San Diego, CA (US). https://www.omgwiki.org/MBSE/lib/exe/fetch.php?media=mbse:patterns:science_math_foundations_for_systems_and_systems_engineering--1_hr_awareness_v2.3.2a.pdf.
- Schindel, W., and R. Dove. 2016. "Introduction to the Agile Systems Engineering Life Cycle MBSE Pattern." INCOSE 2016 International Symposium, Edinburgh, GB-SCT. https://www.omgwiki.org/MBSE/lib/exe/fetch.php?media=mbse:patterns:is2016_intro_to_the_aselcm_pattern_v1.4.8.pdf.
- 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." 2011 Conference of the American Society for Engineering Education (ASEE), Vancouver, CA-BC.
- Schindel, W. and T. Peterson. 2016. "Introduction to Pattern-Based Systems Engineering (PBSE): Leveraging MBSE Techniques." INCOSE 2016 Great Lakes Regional Conference, Mackinac Island, US-MI. https://www.omgwiki.org/MBSE/lib/exe/fetch.php?media=mbse:patterns:pbse_tutorial_glrc_2016_v1.7.4.pdf.
- Schindel, W., and M. Seidman. 2021. "Applying Digital Thread Across the Product Life Cycle." In Defense Network Technical Interchange Meeting, 9-10 June, Indiana Defense Network, Indianapolis, US-IN. https://www.omgwiki.org/MBSE/lib/exe/fetch.php?media=mbse:patterns:team_top_gun_idn_presentation_06.09.2021_v2.1.1.pdf.
- Walden, D. ed. 2015. *Systems Engineering Handbook: A Guide for System Life Cycle Processes and Activities*. Fourth Edition. Revised by D. Walden, G. Roedler, K. Forsberg, R. Hamelin, and T. Shortell. INCOSE, San Diego, US-CA: Published by Wiley.

ABOUT THE AUTHOR

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. He is an INCOSE fellow, chair of the MBSE Patterns Working Group of the INCOSE/OMG MBSE Initiative, and was a member of the lead team of the INCOSE agile systems engineering life cycle discovery project. Bill co-led a 2013 project on systems of innovation in the INCOSE System Science Working Group.

Schindel continued from page 16

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. He is an INCOSE fellow, chair of the MBSE Patterns Working Group of the INCOSE/OMG MBSE initiative and was a member of the lead team of the INCOSE agile systems engineering life cycle discovery project. Bill co-led a 2013 project on systems of innovation in the INCOSE System Science Working Group.

Requirements Statements Are Transfer Functions: An Insight from Model-Based Systems Engineering

William D. Schindel, schindel@ictt.com

■ **ABSTRACT**
Traditional systems engineering pays attention to careful composition of prose requirements statements. Even so, prose appears less than what is needed to advance the art of systems engineering into a theoretically based engineering discipline comparable to electrical, mechanical, or chemical engineering. Ask three people to read a set of prose requirements statements, and a universal experience is that there will be three different impressions of their meaning. The rise of model-based systems engineering might suggest the demise of prose requirements, but we argue otherwise. This paper shows how prose requirements can be productively embedded in and a valued formal part of requirements models. This leads to the practice-impacting insight that requirements statements can be non-linear extensions of linear transfer functions, shows how their ambiguity can be further reduced using ordinary language, how their completeness or overlap more easily audited, and how they can be "understood" more completely by engineering tools.

SYSTEMS ENGINEERING PROSE

Traditional Requirements Discipline. Composing good requirements statements prose has a long tradition in systems engineering. As described in (Buede 2000), systems engineers are typically instructed that effective requirements statements should be:

- Unambiguous
- Understandable
- Correct
- Concise
- Traceable, Traced
- Design Independent
- Verifiable
- Unique
- Complete
- Consistent
- Comparable
- Modifiable
- Attainable.

The resulting requirements describe systems, are stored in databases, expressed in requirements documents, and interpreted by people. (See Figure 1.)

Growing Challenges to Prose. Well-structured prose requirements statements are of course more effective for this worthwhile and practical care. But are traditional prose requirements compositional principles effective enough for the future demands of systems engineering, as it strives to move from a craft-like body of knowledge to a scientifically-founded engineering discipline, comparable to electrical, mechanical, or chemical engineering?

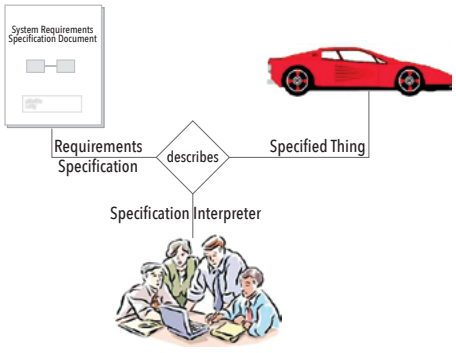


Figure 1. Specification documents describe things to interpreters

Engineered target systems are becoming more complex, more mission-critical, more risk-averse, more in need of clear

human understanding, demanding of faster development cycles, and supported by larger engineering teams, who are interoperating with more engineering tools. Is it reasonable to expect that well-composed prose will be up to these rising technical demands?

Is good prose the bulwark of good engineering, or of good literature? If Newton and his followers had been forced to use only prose to express their reasoning, would today's engineers be using orbital mechanics to design mission systems? The *Principia* (Newton 1668) straddles the transition from prose-based geometric proof to the power of mathematics. Should we expect that our systems engineering prose will be replaced with mathematical equations?

SYSTEMS ENGINEERING MODELS—REPLACEMENT OR EXTENSION?

Model-Based Systems Engineering. INCOSE has helped to lead progress to model-based methods for systems engineering (INCOSE MBSE 2004). The use of graphical and other forms of models has appeared in systems engineering through application of graphically focused modelling languages (backed by underlying information metamodels) such as IDEF0, UML®, and most recently SysML® modelling languages, described in Buede (2000), Oliver (1999), Booch et al. (1999), SysML (2004), and AP233 (2004).

Models are data structures (often, but not always, represented graphically) intended to *explicitly* represent facts (for example, requirements, designs) about systems (refer to Figure 2). These facts might otherwise be *implicit* in knowledge of experienced systems engineers or domain experts. Without explicit models, these facts are not equally obvious to all, require reading "between the lines", and are opportunities for misunderstandings, engineering process errors, rework, or failures.

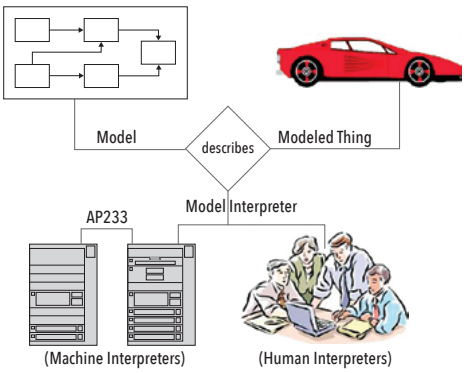


Figure 2. Models describe things to interpreters

The model-based engineering approach, not fully reviewed here, intends that these models, when compared to earlier approaches:

- Are more explicit
- Are more compact
- Enhance visualization, understanding, and communication
- Enhance the formal underlying theoretical structure of engineering information, to improve ability to analyze, simulate, (execute the model), and even synthesize
- Enable database-driven processes instead of document-driven processes
- Improve shared "understanding" of the meaning of models to be exchanged between machines (engineering tools) and humans
- Provide theoretical foundation that was not available in earlier prose-based approaches, supplementing intuition with discipline.

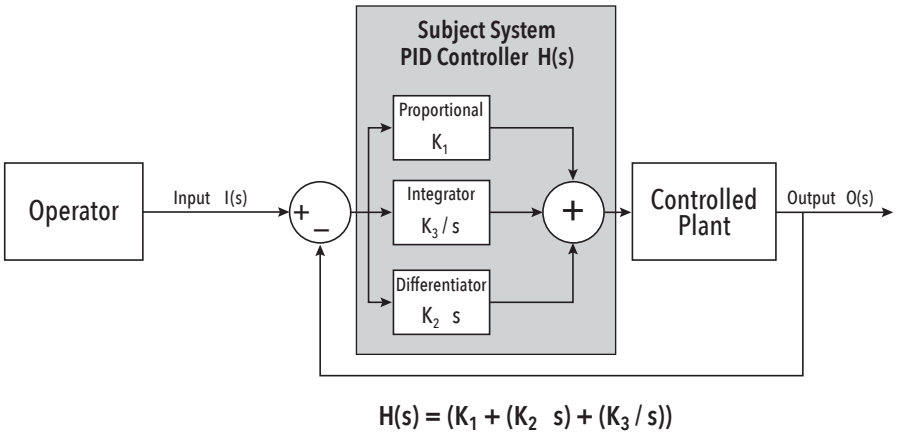


Figure 3. Logical architecture of a PID controller

Electrical schematic diagrams and mechanical drawings, if drawn according to learned disciplines, provide engineers with relatively unambiguous models of systems, compared to the typically more ambiguous experience with prose-based systems engineering requirements documents (Glasgow et al. 1995). Not all diagrams are unambiguous, as famously illustrated by Escher (1992). How can general system requirements be expressed as clearly and unambiguously as the design communicated by an electrical schematic diagram?

Will Diagrams Replace Prose? Progress occurring with model-based systems engineering might lead to the expectation that the graphic components of models will eventually replace the use of prose-based requirements statements altogether. We argue differently here, in spite of the claim that "a picture is worth a thousand words".

Our experience is that the most productive outcome is not the total replacement of prose with diagrams, but a merging of these two forms of information, into a total formal model that includes both. Current efforts to incorporate prose into models in some fashion are described in SysML Partners (2004) for SysML modelling and in AP233 (2004) for the related AP233 activity. We will describe the embedding of prose into the model, as a first-class part of that model. The approach we will describe is something more than just embedding requirements prose as unstructured text. Our inspiration for how prose should be embedded in models comes from examining the underlying meaning of the original requirements prose—the special semantics of requirements statements as a specialized subset of all prose.

Given current practices, tools, trends, and standards efforts, this outcome is by no means obvious and requires both careful theoretical and practical review to understand what is possible and how one might interpret current evolution of practice.

EMBEDDING THE PROSE IN THE MODELS

Transfer Functions. A rich vein in the theory of linear systems is the idea of *transfer functions*, as described in Churchill (1958). Transfer functions describe the relationship between system inputs and outputs, typically in the frequency domain (see Figure 3).

Such a transfer function, parameterized by attributes (K1, K2, K3), completely characterizes the behavior of the associated linear system—all its behavioral characteristics (whether in the frequency or time domain) can in principle be derived from the transfer function description. Unfortunately, this utility appears limited, as most aspects of the systems encountered by engineers are non-linear, and therefore not described in this way.

Nevertheless, we retain one idea from linear systems and their transfer functions—the benefit of characterizing the external behavior of a system in the form of

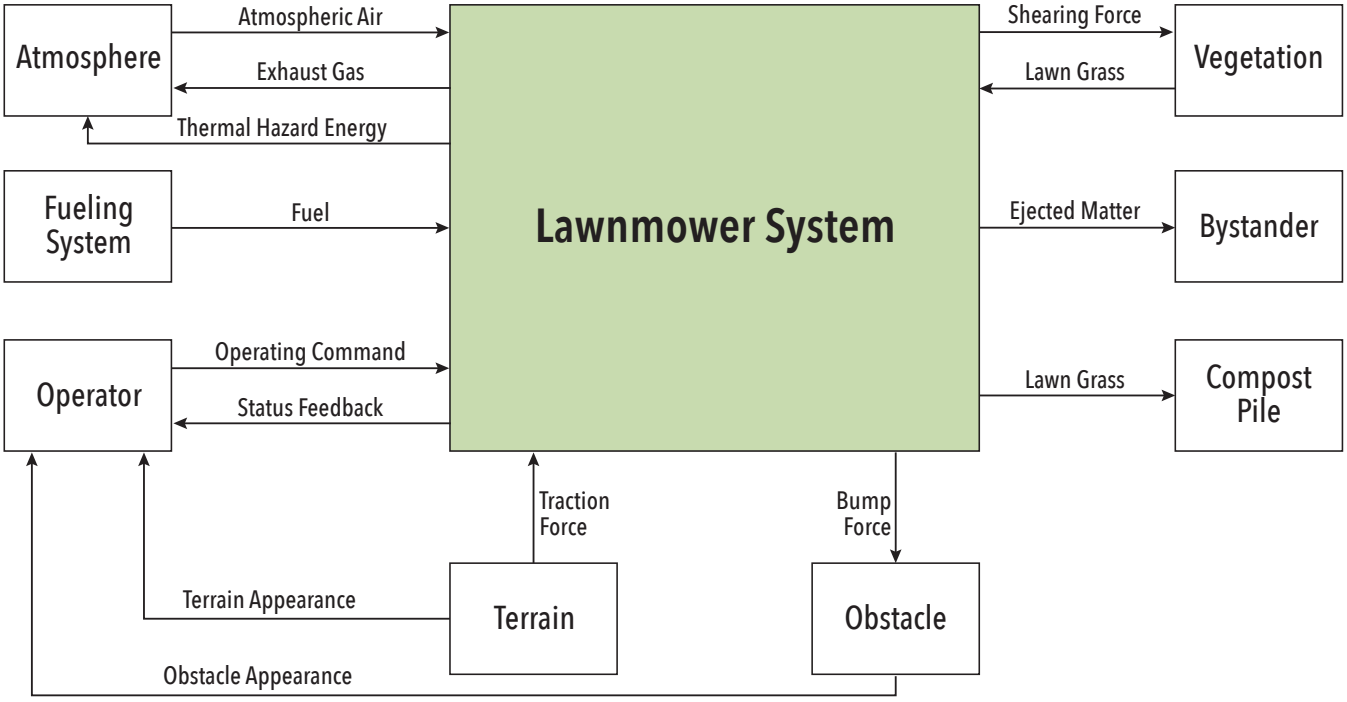


Figure 4. System external inputs and outputs

relationships between its inputs and outputs. Engineers recognize the value of somehow characterizing a system's externally visible behavior, whether in the form of data sheets, commercial specifications, or otherwise. But can we expect to do this in the form of algebraic or differential equations? Most (perhaps all) engineered systems evade practical description in the form of such equations—nonlinear or otherwise. Furthermore, system stakeholders often don't speak the language of mathematics.

What we are interested in retaining (and abstracting) is the idea of *statements of relationships* between inputs and outputs, whether in the form of simple mathematical equations or something else. The idea is to describe the relationship of values of the system's outputs to the values of its inputs—including the impact of time history. Why is characterizing such behavior so important?

What are Functional Requirements, Really? Traditional systems engineering teaches us that requirements are to describe what a system should do, not how it does it. Just what does this really mean, in the language of science, engineering, and mathematics? Whether we use prose requirements statements or some other form of description, the need is to describe the system as a "black box"—describing its required behavior as "seen" by the other systems with which it interacts (Figure 4), without discussing its internal design implementation. This certainly sounds like a formal characterization of the system and could be described in terms of relationships between its inputs and output.

What About "Non-Functional" Requirements? Why don't we typically think of "requirements" as this kind of formal external characterization? At least two issues usually suggest that "requirements" for a system might not be exactly the same idea as this formal behavioral characterization of the system:

1. Stakeholder requirements in the non-technical language and perspective of stakeholders may seem different than such technical system characterization—witness, for example quality function deployment's (QFD's) approach, as described in Clausing et al. (1988).
2. Requirements for system reliability, manufacturability, supportability, or the other "ilities," as well as system capacities, tolerances, or other requirement categories may seem different than system technical characterization—to the point that these are sometimes referred to as "non-functional" requirements, as in Chung et al. (1999).

However (as discussed later below), all these different needs eventually *map to and can be productively expressed as* the external behavior of the system, expressed as the totality of its input-output relationship characteristics. Indeed, for all of these, systems engineers typically "derive" technical language requirements that are in principle objectively testable (or otherwise analyzed) input-output behaviors at the interfaces of the system, while recognizing that such "technical requirements" are different (transformed from) the original stakeholder requirements.

The theme of our argument is that all requirements statements are input-output relationships, or generalizations of transfer functions, expressing external behavior. (Design constraints are not a part of this argument, but are likewise traceable to desired external behaviors, which should be included.) A key attraction of this view is to take better advantage of the edifice of the scientific- mathematical understanding based on physical interactions that was built by giants such as Newton, Hamilton, Maxwell, and others (Newton 1668, Hamilton 1834, Sussman 2001, and Landau et al. 1976), as the scientific underpinnings of modern engineering disciplines. Indeed, the overall system engineering methodology from which this view is taken (Schindel 2002, Schindel et al. 2002) is based upon the concept of interactions of multiple components, versus the behavior of any single component in isolation. While this viewpoint is foundational in science, systems engineering of requirements sometimes takes the isolated system perspective, leading to later integration challenges.

We treat all the requirements on a system as ultimately expressed (possibly through derivation mappings from stakeholder needs or other forms) in the form of system interaction behavior at external system boundaries. This behavior can be described in terms of relationships between system inputs and outputs that characterize the system. These relationships are frequently

parameterized by system attributes. This also allows us to take advantage of both contemporary modeling languages (SysML Partners 2004, AP233 2004), as well as the success of physics.

The Return of Prose to the Model. The approach described here, then, is to see requirements statements prose as existing solely for the purpose of expressing required system input-output behavioral relationships. This is not the most widely held or traditional view of requirements statements. We can conceive of our typical prose sentences as a kind of generalization of mathematical equations, expressing what in the end really are relationships between input and outputs. Indeed, this approach is familiar in the world of VHSIC hardware description language (VHDL) characterization of digital electronics (Ashenden 1996) or propositional calculus assertions about system logical behavior (Carnap 1958). It also fits the cases in which there are algebraic, differential, or integral equations relating inputs and outputs mathematically (whether deterministically or stochastically), as well as fuzzy relationships (Zadeh 1965). Similarly, requirements statements may describe relationships by the use of tables, graphs, or other representations of input-output (I/O) relationships. Developers of modeling languages benefit from noting that equations, graphs, tables, may not need improvement, but instead direct incorporation into the model. But what is the practical implication for regular prose sentences in English or other national languages?

The first implication of this approach is a simplified way to think about requirements statements: As shown in Figure 5, requirements statements for a given subject system need only contain:

1. References to inputs and outputs.
2. Statements of relationships between them, including attributes that parameterize those relationships.

This simplifies the process of composing, as well as interpreting, analyzing, and auditing, these requirements statements. It does so by more than traditional prose grammatical means—it does so by taking a mathematical—physical modeling view of systems, in the tradition of engineering and science. At the same time, it preserves the "normal everyday language" aspects of requirements statements.

Figure 5 is an information model of prose requirements statements, and some such statements will contain multiple (or no) instances of the classes it describes. While not every requirement statement needs to contain inputs, outputs, or attributes, every such statement should contain at least one of these, and will refer to at least one relationship. There should always be a clear association to a subject system. The complete characterization of the total relationship between system inputs and outputs is the union of all that system's prose requirements statement models.

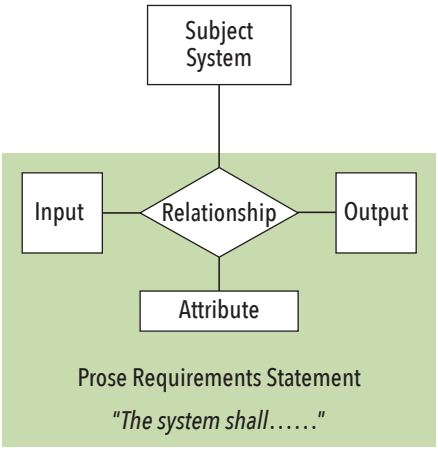


Figure 5. Prose requirements metamodel

In the following example requirements statements, the prose has been punctuated to make the components of Figure 5 more evident:

- Inputs and outputs are underlined.
- [Attributes] are in brackets.
- Relationships are italicized.

1. "The Lawnmower System shall *operate* with [Hourly Mowing Capacity] of at least 1 level ground acre per hour, at [Max Elevation] up to 5,000 feet above sea level, and [Max Ambient Temperature] of up to 85 degrees F., at up to 50% [Max Relative Humidity], for [Foliage Cutting Capacity] of Acme American Standard one week Lawn Grass."
2. "The Lawnmower System shall *operate* using Fuel consisting of gasoline having a [Min Octane Rating] of not less than 92, combusted with Atmospheric Air."
3. "The Lawnmower System shall operate with [Fuel Economy] of at least 1 hour / gallon at [Min Elevation] of 0 feet ASL, at [Max Ambient Temperature] 85 degrees F., 50% [Max Relative Humidity], for Acme American Standard one week Lawn Grass."
4. "The Lawnmower System shall *operate* with [Elevation Derating] of 10% improvement in [Fuel] per 1,000 feet of elevation reduction, to a [Min Elevation] of 0 feet ASL.
5. "The Lawnmower System shall *operate* meeting the more demanding of state and federal standards for [Max Gaseous Pollution] and [Max Particulate Pollution] of Exhaust Gas.
6. "The Lawnmower System shall *operate* with [Operating MTBF] no less than 500 hours."
7. "The Lawnmower System shall *operate* so as to protect the Operator from Thermal Hazard Energy by maintaining all accessible metallic surfaces at a [Maximum Surface Temperature] of less than 180 degrees F."

Decomposition, Logical Architecture, and Allocation. Prose requirements statements are traditionally transformed into derived statements that describe requirements having smaller scope and/or greater specificity or detail. This traditional approach is matched in the perspective described here by the process of decomposing (partitioning) a subject system into *logical subsystems*. These are groupings of required external behavior, not design allocations. Refer to Figure 6 for a Logical Architecture partitioning external behavior (not design) of Figure 4.

In the above example, requirement (1) is decomposed in the same way that Figure 6 decomposes the Lawnmower System into subsystems. Each of the following requirements, derived from requirement (1) above, is allocated to a different logical subsystem of Figure 6:

1.1) "The Power Subsystem shall *generate* [Power Output] of combined Propulsion Power and Cutting Power at [Max Elevation] up to 5,000 feet above sea level, and [Max Ambient Temperature] of up to 85 degrees F., at up to 50% [Max Relative Humidity]."

1.2) "The Carriage and Drive Subsystem shall generate a Traction Force sufficient to *propel* over an [Hourly Mowing Capacity] of at least 1 level ground acre per hour, by converting a Propulsion Power input of [Traction Power Consumption]."

1.3) "The Cutting Subsystem shall generate Shearing Force sufficient to *cut and capture* Lawn Grass of at least 1 ground acre per hour of [Foliage Cutting Capacity] of Acme American Standard one week Lawn Grass, by converting a Cutting Power input of [Cutting Power Consumption]."

This illustrates the introduction of intermediate (internal) input-output variables, as well as subsystem attributes and budgets.

ADDITIONAL IMPLICATIONS

The above argument leads us to write

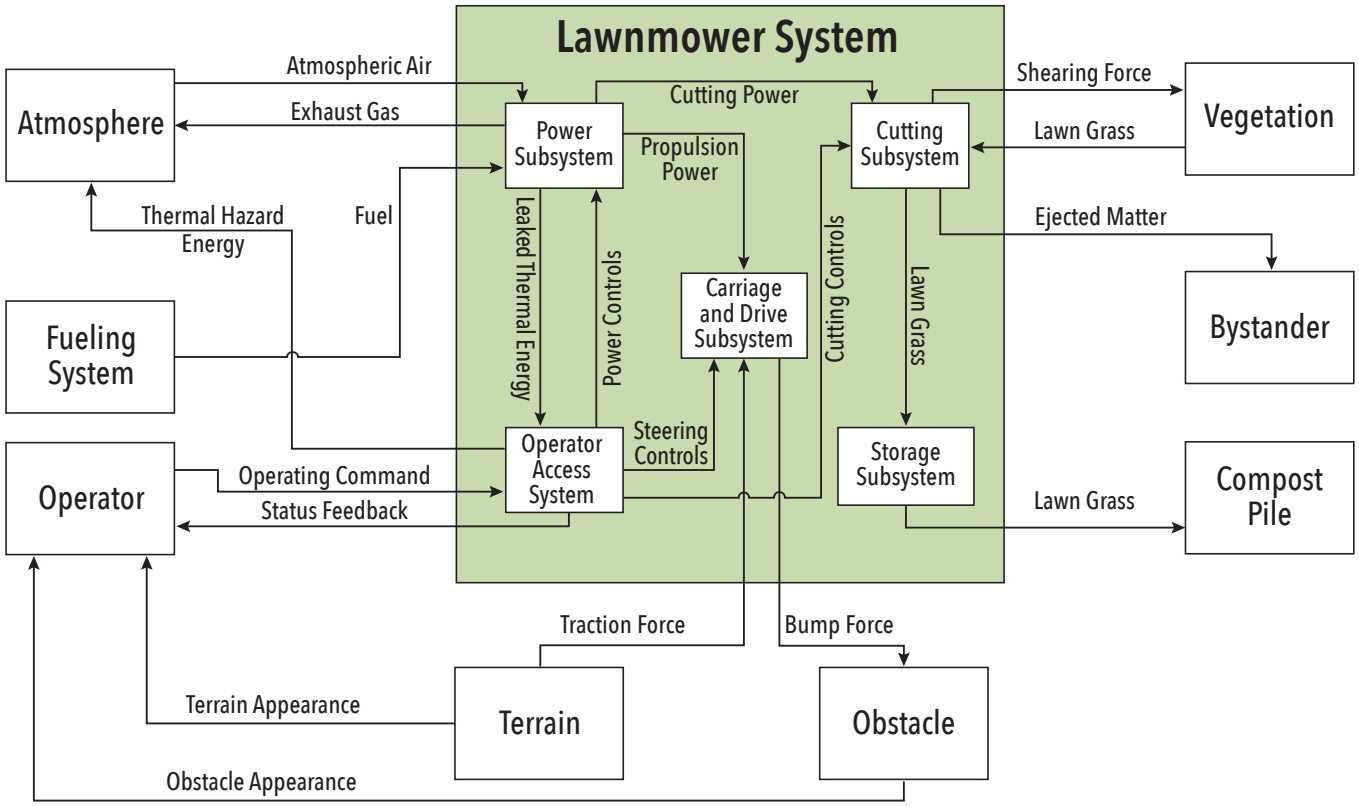


Figure 6. Logical architecture – partitioning of Figure 4

technical requirements statements (derived from less technical statements of stakeholder needs) that explicitly describe (parameterized) input-output relationships. This has a number of additional implications, including the following areas.

Improved, Inspectable Statement Structure. Expressing requirements in the form described here significantly improves the ability to inspect requirements for completeness, as well as clarity. This is because they are expressed with respect to (and in fact are embedded in and part of) a system model, in which their formal role is now exactly to characterize system outputs with respect to system inputs. We can ask, "Does this set of requirements statements specify the required system output values for all input values (and histories)?" Lack of coverage, as well as overlap or conflict, are more easily detected. A traditional challenge of specifying system requirements is understanding whether we are "done" generating them. While some of this is uncertainty about stakeholder needs, in more complex systems a typical problem is determining whether a candidate requirements specification completely characterizes the system at all.

Implications for Tools. When the micro-structure of requirements statements is understood to be a parameterized statement of relationship between inputs and outputs, then tools can better "understand" the semantics of these statements. Such tools can potentially answer questions about requirements, even to the extent of improving the executability of models in simulation, a goal described in Mellor (2002) and Karayanakis (1993). Simulation relationships are directly embedded in the model.

Implications for Models. When the micro-structure of requirements statements is understood to be a parameterized statement of the relationship between inputs and outputs, then the form of requirements models themselves can be more expressive and explicit than if these statements are simply viewed as strings of textual prose. Refer to the requirements diagrams of SysML Partners (2004).

Implications for Reuse. By making the key variables of requirements explicit parameters (attributes) of requirements statements, we have improved not only our understanding of key parametric relationships, but also the configurability of those requirements for re-use. Reusable designs and design platforms arise from the re-usability of requirements, enhanced by this approach. This is discussed further below.

Relational-Symbolic Duality. This approach illustrates the underlying duality of the representation of requirements in symbolic (for example, prose or equations) versus relational (for example, graphical or relational models) form, as further discussed in Schindel (1997), Hayakawa (1990), Chomsky and Piaget (1980), Whorf (1956), and Marcus (2001).

WHAT IS LEFT? THE NON-PROSE PART OF REQUIREMENTS MODELS

In this paper and the systems engineering methodology it references (Schindel et al. 2002) we consider prose requirements statements to be a part of, but not all of, a total model of requirements. The referenced methodology grew out of research seeking the minimal information necessary to describe a system (Schindel 2002). Having clarified above the role of prose requirements statements as one formal part of a total requirements model, it of interest to ask: What is the rest of the requirements model? Why is more needed than the requirements prose alone, if such prose describes all the I/O relationships?

Additional Metamodel Components. While this issue would ultimately take us on a trip through modelling that goes beyond the scope of this paper, it is interesting to briefly see the roles of the rest of a requirements model with respect to that of the prose requirements statements alone. The following additional model components, summarized by Figure 7 and described in Schindel (2002) are also needed to complete the requirements model, and they answer the listed requirements

questions, supplementing the requirements statement prose:

1. Domain Model: What is environment of the subject system? With what external systems (actors) does it interact, through what external interfaces? What are the key relationships of this domain? What external interactions are eligible to be characterized by, and must be covered by, prose requirements statements?
2. Stakeholders and Needs Model: What are the primary stakeholder roles played by people or organizations with a stake in the system, and what are their (voice of the stakeholder) needs?
3. Feature Model: How are the behaviours of the system organized with respect to the values of its stakeholders? What attributes describe these values in stakeholder terms? These are the stakeholder language behaviours eligible and required to be technically characterized by requirements statement prose.
4. State Model: How do the behaviour functional relationships between the subject system and its environment change modalities in the presence of different environmental situations (states)? What is the temporal model of the environment, and when (with respect to that temporal situational model) do different requirements apply?
5. Functional Interaction Model: What is the organization of the technical physical interactions of the system with its environment? How are these interactions described by the input-output relationships described by the prose requirements? What are the key technical attributes describing this behaviour, and how are these coupled to the stakeholder value-based feature attributes?
6. Logical Architecture Model: How is the externally-viewed functional interactive behaviour of the system decomposed and organized by partitioning it into a logical architecture of behaviour? This partitioning describes the decomposition and derivation of lesser scope detailed requirements.

These additional requirements model components provide context and fill out the formal meaning of the requirements statements that are embedded into them. For example, the connection of the requirements statements to the state model of (4) above illustrates the embedding recently described in Daniels and Bahill (2004).

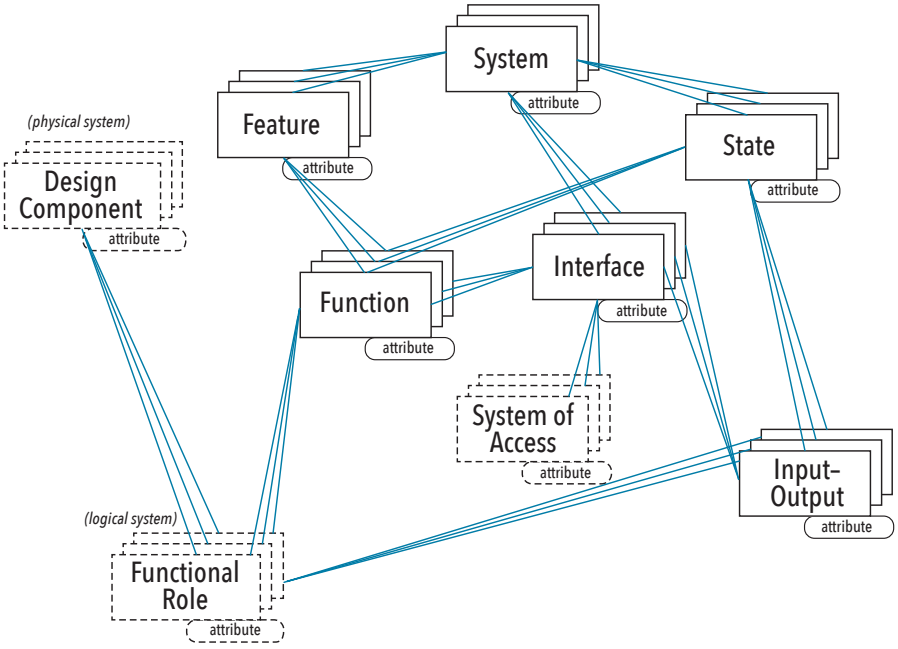


Figure 7. Summary of the larger systems engineering metamodel

Still other components of this metamodel relate these requirements to system design, verification, etc. This overall metamodel is related to, although more abstracted than, the AP233 metamodel described in AP233 (2004).

PATTERNS: REUSABLE, CONFIGURABLE MODELS OF REQUIREMENTS

Reusable designs are possible only because of reusable requirements — some commonality of needs across different market segments, customers, applications, product lines, or sub-systems. Patterns are re-usable models. While first popularized in some domains for *design* patterns (Gamma et al. 1995, Alexander et al. 1977), they are of interest as patterns of *requirements*.

The embedding of parameterized requirements statements in overall requirements models described in this paper, along with the other aspects of the object oriented metamodel of Figure 7, create a modelling framework that enables pattern-based systems engineering (Schindel 2002, Schindel and Smith 2002). This approach introduces patterns as re-usable models, cast in the metamodel of Figure 7.

Using this approach, Figure 8 illustrates the process by which patterns of requirements and designs for generic systems can then be configured or specialized into individual product line families, and ultimately individual product systems. This approach has been applied in several commercial off the shelf (COTS) product line enterprises, to enhance COTS portfolio engineering and planning. This approach also facilitates the ongoing expression of organizational learning in the form of updates and refinements to "uncovered" patterns. A particularly striking benefit of this approach is that it allows large organization practitioners who are less skilled in "clean sheet" original modelling to gain the benefits of model-based engineering. This is accomplished by teaching larger groups the generic system pattern models of the enterprise, for their configuration and use. We have found this is more easily learned by larger groups than abstract modelling methodologies.

RESULTS AND CONCLUSIONS

In conclusion:

1. Prose requirements statements have an important role to play as a part of future model-based requirements data structures, as generalizations of transfer functions. This unifies a traditional requirements-writing skill with emerging model-based engineering techniques.
2. Requirements statements can be written in every-day natural language to explicitly refer only the system inputs, outputs, relationships between them, and parametric attributes of those relationships.
3. This improves the ability to write, understand, inspect, and use prose requirements statements, and improves the usual discipline of writing requirements statements, while maintaining traditional principles of requirements.
4. This approach also unifies the incorporation of requirements prose with

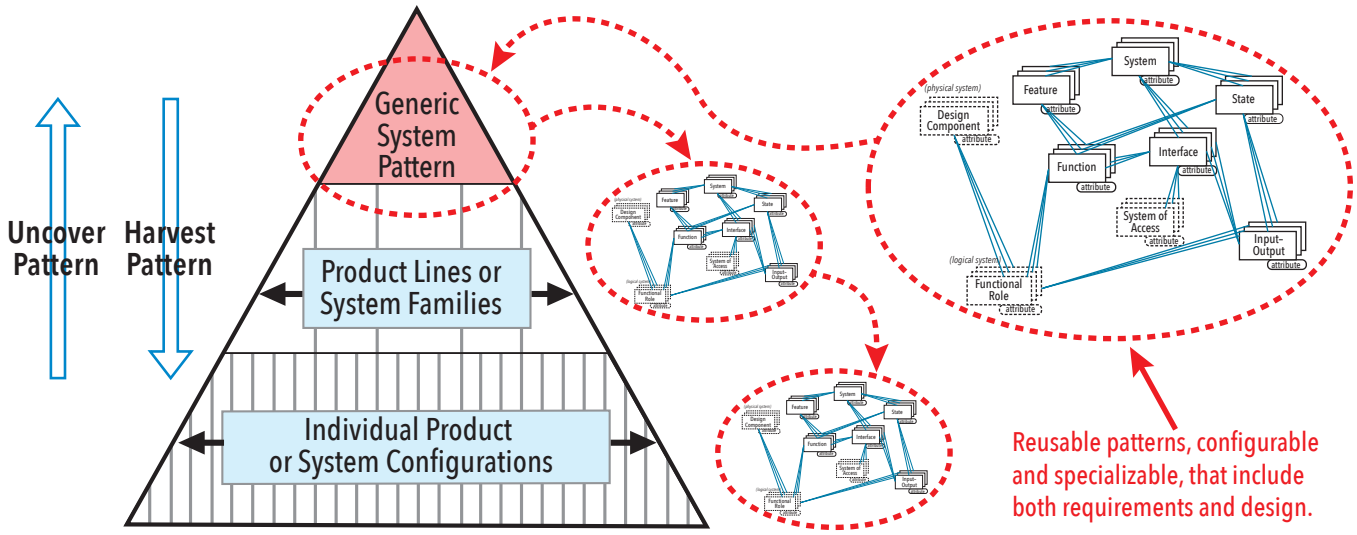


Figure 8. Patterns are configurable, re-usable models of requirements and designs

other forms of input-output relationships, including equations, tables, graphs, and other relations.

5. This approach also improves the ability to create requirements patterns—libraries of configurable, re-usable requirements, improving the performance of the engineering process across larger product line and COTS enterprises.
6. Automated modelling and requirements tools can increase in their capabilities using this paradigm. We have applied this approach using the systems engineering and modelling tools of a number of tools suppliers.
7. Less experienced engineers can apply these concepts to improve their requirements writing and modelling. We have successfully taught this approach to undergraduate and graduate engineering students, as well as practicing engineers in commercial and mil-aero organizations. ■

REFERENCES

- Alexander, C., S. Ishikawa, I. Fiksdahl-King, and S. Angel. 1977. *A Pattern Language: Towns, Buildings, Construction*. New York, US-NY: Oxford U. Press.
- AP233 (ISO 10303). 2004. Web Site, https://step.nasa.gov/.
- Ashenden, Peter J. 1996. *The Designer's Guide to VHDL*., San Francisco, US-CA: Morgan Kaufmann Publishers, Inc.
- Booch, G., J. Rumbaugh, and I. Jacobson. 1999. *The Unified Modelling Language User Guide*. Reading, US-MA: Addison Wesley.
- Buede, Dennis M. 2000. *The Engineering Design of Systems: Models and Methods*. New York, US-NY: John Wiley & sons, Inc.
- Carnap, Rudolf. 1958. *Introduction to Symbolic Logic and Its Applications*. New York, US-NY: Dover Publications.
- Chomsky, N., J. Piaget, et al. 1980. *Language and Learning: The Debate Between Jean Piaget and Noam Chomsky*. Edited by Massimo Piattelli-Palmarini. Cambridge, US-MA: Harvard University Press.
- Chung, L., B. A. Nixon, E. Yu, and J. Mylopoulos. 1999. *Non-Functional Requirements in Software Engineering*. Springer Publishing.
- Churchill, Ruel V. 1958. *Operational Mathematics*. New York, US-NY: McGraw-Hill Book Company.
- Clausing, D., and R. Hauser. 1988. "The House of Quality." *Harvard Business Review*, MayJune.
- Daniels, J., and T. Bahill. 2004. "The Hybrid Process That Combines Traditional Requirements and Use Cases." *Systems Engineering* 7 (4): 303-319.
- Escher, M. C. 1992. *M. C. Escher: The Graphic Work*. Benedikt Taschen Verlag Publishers.
- Gamma, E., R. Helm, R. Johnson, and J. Vlissides. 1995. *Design Patterns: Elements of Reusable Object-Oriented Software*. Reading, US-MA: Addison-Wesley.
- Glasgow, J., N. Hari Narayanan, and B. Chandrasekaran, eds. 1995. *Diagrammatic Reasoning: Cognitive and Computational Perspectives*. Cambridge, US-MA: MIT Press.
- Hamilton, William Rowan. 1834. "On A General Method in Dynamics." *Philosophical Transactions of the Royal Society* Part II: 247-308.
- Hayakawa, S. I. 1990. *Language in Thought and Action*. Fifth Edition. New York, US-NY: Harcourt Brace Jovanovich.
- INCOSE MBSE. 2004. INCOSE Model Driven System Design Working Group web site, http://www.incose.org/practice/techactivities/modelingtools/mdsdwg.aspx.
- Karayanakis, Nicholas M. 1993. *Computer-Assisted Simulation of Dynamic Systems with Block Diagram Languages*. Boca Raton, US-FL: CRC Press, Inc.
- Landau, L. D., and E. M. Lifshitz. 1976. *Mechanics*. Third Edition. From *Course of Theoretical Physics, Volume 1*. Oxford, GB: Elsevier Science Ltd.
- Marcus, Gary F. 2001. *The Algebraic Mind: Integrating Connectionism and Cognitive Science*. Cambridge, US-MA: MIT Press.
- Mellor, S., and M. J. Balcer. 2002. Executable UML: *A Foundation for Model-Driven Architecture*. Boston, US-MA: Addison-Wesley.
- Newton, Isaac. 1668. *Mathematical Principles of Natural Philosophy*. trans. and ed. by Andrew Motte, 1729; rev. by Florian Cajori, 1934 Berkeley, US-CA: University of California Press, Berkley, US-CA.
- Oliver, David. 1999. *Engineering Complex Systems*. New York, US-NY: McGraw-Hill.
- Pinker, Steven. 1994. *The Language Instinct: How the Mind Creates Language*. New York, US-NY: William Morrow & Co.
- 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, November.
- Schindel, William D. 2022. "Does Our SE House Have a Foundation?" INCOSE Crossroads of America Chapter technical program presentation, Peoria, US-IL, 22 May.
- Schindel, William D. 1996. "System Engineering: An Overview of Complexity's Impact." SAE International, Technical Paper 962177, October.
- Sussman, G. J., and J. Wisdom. 2001. *Structure and Interpretation of Classical Mechanics*. Cambridge, US-MA: MIT Press.
- SysML Partners. 2004. Web Site, http://www.sysml.org/.
- Whorf, Benjamin Lee. 1956. *Language, Thought, and Reality: Selected Writings of Benjamin Lee Whorf*. John B. Carroll, editor. Cambridge, US-MA: MIT Press.
- Zadeh, L. A. 1965. "Fuzzy Sets." *Information and Control* 8: 338-353.

ABOUT THE AUTHOR

William D. Schindel is president of ICTT, Inc., a systems engineering company, and the developer of the Systematica™ methodology for model and pattern-based systems engineering. His 35-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, 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.

Feelings and Physics: Emotional, Psychological, and Other Soft Human Requirements, by Model-Based Systems Engineering

William D. Schindel, schindel@ictt.com

■ **ABSTRACT**
Traditionally, engineering encourages requirements statements that are objective, testable, quantitative, atomic descriptions of system technical behavior. But what about "soft" requirements? When products deliver psychologically or emotionally based human experiences, subjective descriptions may frustrate engineers. This challenge is important for products appealing to senses of style, enjoyment, fulfillment, stimulation, power, safety, awareness, comfort, or similar emotional or psychological factors. Automobiles, buildings, consumer products, packaging, graphic user interfaces, airline passenger compartments and flight decks, and hospital equipment provide typical examples. This paper shows how model-based systems engineering helps solve three related problems: (1) integrating models of "soft" human experience with hard technical product requirements, (2) describing how to score traditional "hard" technology products in terms of "fuzzier" business and competitive marketplace issues, and (3) coordinating marketing communication and promotion with the design process. The resulting framework integrates the diverse perspectives of engineers, stylists, industrial designers, human factors experts, and marketing professionals.

"SOFT" HUMAN REQUIREMENTS: THE ENGINEER'S CHALLENGE

Human-Experienced Qualities. Traditional engineering methods encourage us to write requirements statements that are objective, testable, quantitative, atomic descriptions of desired system technical behavior. It has been shown (Schindel 2005) that such requirements prose may be directly generated by model-based methods. This paper explores the opposite direction: Requirements for products and systems that interact with people are frequently expressed in terms of human-experienced qualities. For some system products (for example, aircraft passenger compartments, furniture, tools, entertainment systems, clothing), these may be among their most important requirements. For other products (for example, control systems, manufacturing processes, buildings), these requirements can be at least a critical subset of the total requirements.

The descriptions of such "soft" qualities often use nomenclature and ideas of psychology, emotion, and other human-based terminology, and may originate from non-technical laymen, or from technical specialists who study human nature instead of engineering and physics. This can leave the engineer writing technical product or system engineering specifications in a dilemma. How does one treat seemingly "soft" requirements of this type seriously, link them to technical designs, and subject them to formal and effective validation and verification?

The Challenge to Engineers. These questions frequently lead to uncertainty or frustration on the part of the engineer, or a

sense that requirements of this sort cannot be treated the same as "hard technical" requirements, such as one finds in interactions between non-human systems. How is an engineering-trained designer to accommodate requests that a product should make its human user "excited", "fulfilled", "undistracted", or "uplifted"? How can engineers in such cases feel that their work is conducted in a technically sound, systematic, and optimized fashion?

Human-based requirements of this sort are essential in the design of consumer products, military and commercial aircraft and vehicles (which interact with pilots and operators), therapeutic devices and systems, and many other products. Techniques such as quality function deployment (QFD) (Clausing et al. 1988) and axiomatic design (Suh 2001) express certain relationships about soft requirements, but may do so without fully communicating the human factors specialist's understanding, and their full integration into model-based systems engineering processes is not always clear.

Industrial Designers and Architects. The technical design community is not without success in the design of human-oriented systems. The work of Raymond Loewy (1998), Henry Dreyfuss (Flinchum 1997), Louis Sullivan (1956), Frank Lloyd Wright (Pfeiffer 1993), and other industrial designers and architects reveals a rich heritage of design for human experience. The work of these pioneers illustrates intuitive genius but may not fully reveal a systematic process joining human experiential needs with technical requirements and designs. How does the systems engineer make this connection?

Contributors from Other Fields. The systematic study of human-experienced qualities and related behaviors is the domain of other disciplines originating outside engineering. The modern analytical expression of human psychological systems dates back to at least William James (1950), Sigmund Freud (Hutchins 1952), Carl Jung (DeLaszlo 1993), and their followers, with the introduction of the logical system concepts such as the *unconscious* and *conscious*, or *ego* and *id*, etc. For these pioneers, the systems described did not necessarily have a claimed physical basis—in the terminology of methods described herein, they were "logical" systems not "physical" systems. With the eventual emergence of the disciplines of neuroscience and cognitive psychology in the late twentieth century, researchers such as Domasio (1994), Crick (1994), Edelman (1989), and others explored more deeply the possible physical mechanisms for consciousness, emotion, and their connection to cognitive processes. Studies of the physical basis of human consciousness, emotion, and cognition have most recently moved to the center stage of hard-science sub-disciplines of neuroscience. Arguments about these kinds of logical-physical system associations in humankind are much older. They include the mind-body problem, debated by Descartes (Gaukroger 1995) and others as one of philosophy's central questions. Fortunately for the product design engineer on a commercial schedule, we need not answer these questions of the ages to practice an effective system design approach.

Other related work may be found in model-based systems engineering (AP233 2004, INCOSE MBSE 2004, SysML Partners 2004), quality function deployment (QFD) (Clausing et al. 1988), axiomatic design (Suh 2001), and fuzzy set theory (Zadeh 1965). The framework described here acknowledges and relates these and other conceptual ancestors.

THE APPROACH IN A NUTSHELL

This approach describes an integrated model-based conceptual framework in which product engineers, human factors experts, marketing communication specialists and product planners can work productively together as a team, linking and coordinating their various needs and solutions with improved consistency—while still using different perspectives, tools, and concepts natural to their specialties. We will summarize how "soft" requirements of actual or promoted human experience can be formally described in model based systems engineering (MBSE) models, using a specific systems engineering methodology (Schindel et al. 2002).

This approach uses the MBSE concept of logical systems to represent behavior-based knowledge of softer human dimensions, avoiding the conundrums of the physical basis of mind and experience. The very same MBSE tools are also used to describe the "hard" behavioral requirements of the engineered product with which the human interacts. These two model segments are brought together in a single interacting system domain model, to reveal their interdependencies, consistencies, and inconsistencies. Marketing and human factors specialists can "own" the human part of this model, and product engineers can "own" the technical product part of the model. The resulting unified framework provides a more productive means for these two different professional groups to work together to reach a common understanding of product requirements and opportunities to meet human perceived experience objectives.

In further extensions also described here, the same principles are additionally shown to address other types of "soft" problems that apply even to "hard" technical products: competitive choice, marketable features, product positioning, and promotional programs.

INTRODUCTORY PRINCIPLES AND MODELING TECHNIQUES

Modeling Interacting Systems. The perspective here is that requirements of all types ultimately connect to external physical interactions between systems (Schindel 2005). We say that systems "interact" when they can impact each others' (physical) states, through the (physical) exchange of energy, mass, force, or information (all of which are modeled as "input-outputs").

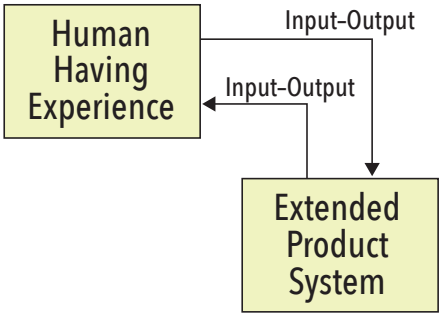


Figure 1. The perspective of human-product interaction

Such a "physics-like" interaction perspective is summarized at the most abstract level by Figure 1, in which the interacting systems are defined as follows:

1. Extended Product System: Includes the subject system, for which we will specify requirements and design. It may be a manufactured product, a service-providing system, or any system. It is called "extended" because it also includes other systems in the product's domain (environment), with which the human and/or product also interacts.
2. Human Having Experience: This is a human being that interacts with the extended product system, for whom we want certain experience-based outcomes to occur.

Modeling "Soft" Qualities of Human Experience. This "physics" oriented model-based strategy for the technical product is extended to soft human experience issues by taking advantage of two key observations:

1. To understand "soft" product requirements based on human emotional or psychological experiences, we must model the *human*, not just the *product*.
2. These models are about externally-observed human behavior, not

the internal physical basis of that behavior—we don't have to understand the physical basis of mind to get the practical results needed for the development process.

This methodology uses the concept of logical systems to model externally visible behavior— including human behavior as well as engineered systems behavior. This approach allows the introduction and use of concepts familiar to the psychologist, but usually considered by the engineer to be "soft" in nature when applied to humans. It then allows these to be linked in an unbroken model chain to hard technical requirements on engineered product interactions.

The following definition is provided by the referenced methodology: A logical system is a system that is defined based upon its externally visible behavior, not its physical identity. "Externally visible behavior" means that which can be "seen" by other systems through physical interactions with an observed system. This means that we can model logical systems without knowing their physical implementation, much as early psychologists (for example, Freud) described theories of human psychological structure without need to describe their physical basis. Figure 2 shows a simplistic model illustrating this approach.

Freud was not required to explain the physical basis of *id* and *ego* in order to use these concepts to advance the description of human psychology. The point here is not whether Freud's early models were "correct" (the reader can substitute a favored model), but rather that these models could be described and externally tested without having to allocate the logical systems of the model to particular physical mechanisms. Such models express the logical architecture of behavior by partitioning that behavior into interacting logical subsystems that are nothing more than components of externally verifiable behavior.

For example, Figure 3 represents the relatively more elaborate ideas that behavior can be partitioned to include:

1. Hierarchical behaviors concerned with basic functioning, higher level planning, aspirations and values (hierarchy of needs was described by Maslow (1962));
2. Attention-focusing systems that regulate the application of finite information processing resources to priority issues (LaBerge 1995);
3. Emotional systems that span multiple levels to regulate behavior globally (Damasio 1994);
4. One's own image of oneself, as well as one's environment—also including how one thinks of the product and one's own use of it. (Trout et al. 1981).

The logical systems shown in Figure 3 accordingly model the following components of externally visible behavior:

1. Sensory Subsystem: Converts "hard" external physical interaction inputs into other representations.
2. Structural and Lower Motor Subsystem: Converts internal signals in the nervous system into external physical output motions or dynamic or static forces.
3. Self-Environment Modeled System: Maintains an internally perceived model of the self (the human's perception of self) and of its interactions with its environment. This environment includes in particular the product system. These logical systems are called "modeled" to differentiate them from the "real" external systems—they are the human's constructed, subjective perceptions of those systems and the self. For some products, the modeled attributes of the self-environment modeled system of Figure 3 include some of the most important customer satisfaction attributes to be supported by the hard technology of the product system.
4. Lower Level Neural Processing: Performs unconscious processing important for regulating bodily processes, survival, and other base functions.
5. Emotion System: Interacts with all levels of conscious and unconscious processes to provide for overall regulation of same.
6. Attention Management System: Manages the resources of conscious level processing to direct limited attention capacity to the highest value perceived situations.

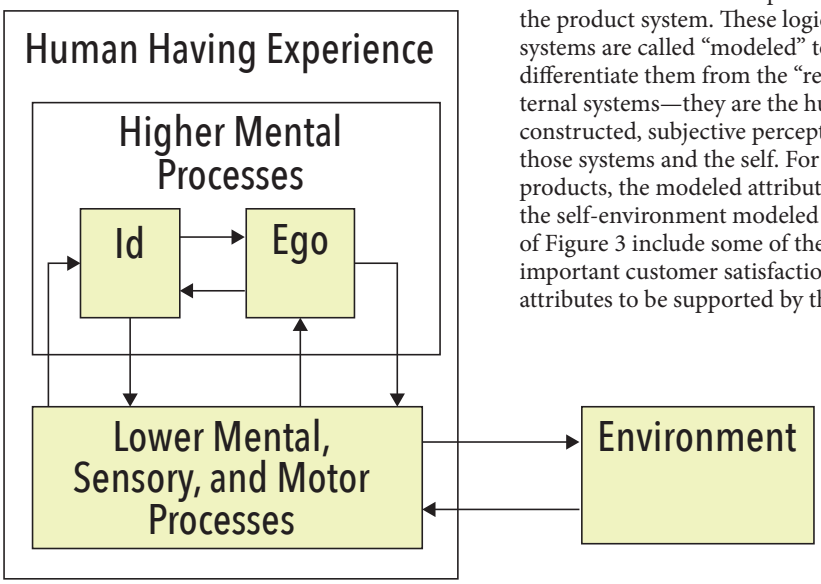


Figure 2. Logical subsystems organize externally visible behavior

The specific model used above is not the main point—those expert in current or specialized psychological models can replace the example shown with their own logical constructs fit to local needs. The key point here is modeling of human behavioural components for integration with product performance—all in a single integrated framework that enables different professionals to work together more successfully.

By the time interactions are shown between the human user and the product system, they include representation of physical input-outputs:

- Information:
  - Visual (appearance)
  - Tactile (feel)
  - Olfactory (smell)
  - Audible (sound)
  - Thermal (heat and cold)
  - Informative Forces (orientation, pressure, acceleration)
- Forces (physical manipulation)
- Mass Transfer (ingestion or secretion/ excretion of mass)
- Thermal Energy (heat transfer).

MODELING THE BEHAVIOR OF THE PRODUCT

A similar approach is used to model the logical architecture of behavior of the product, including its logical subsystems, as shown in Figure 4. It can be seen that Figures 3 and 4 alternatively "telescope" the product versus human behavioral models, ready for integration together.

The model of the product and human behaviors consist of more than just collaboration diagrams of their logical systems. Other parts of the associated meta-model include features, (functional) interactions, states, and interfaces. For purposes of this paper, we will focus on the subset of the meta-model concerned with describing the quantitative relationships between the attributes of the person, engineered system, and environment. This requires an understanding of models of functional interactions, their logical roles, and the quantitative coupling relationships between

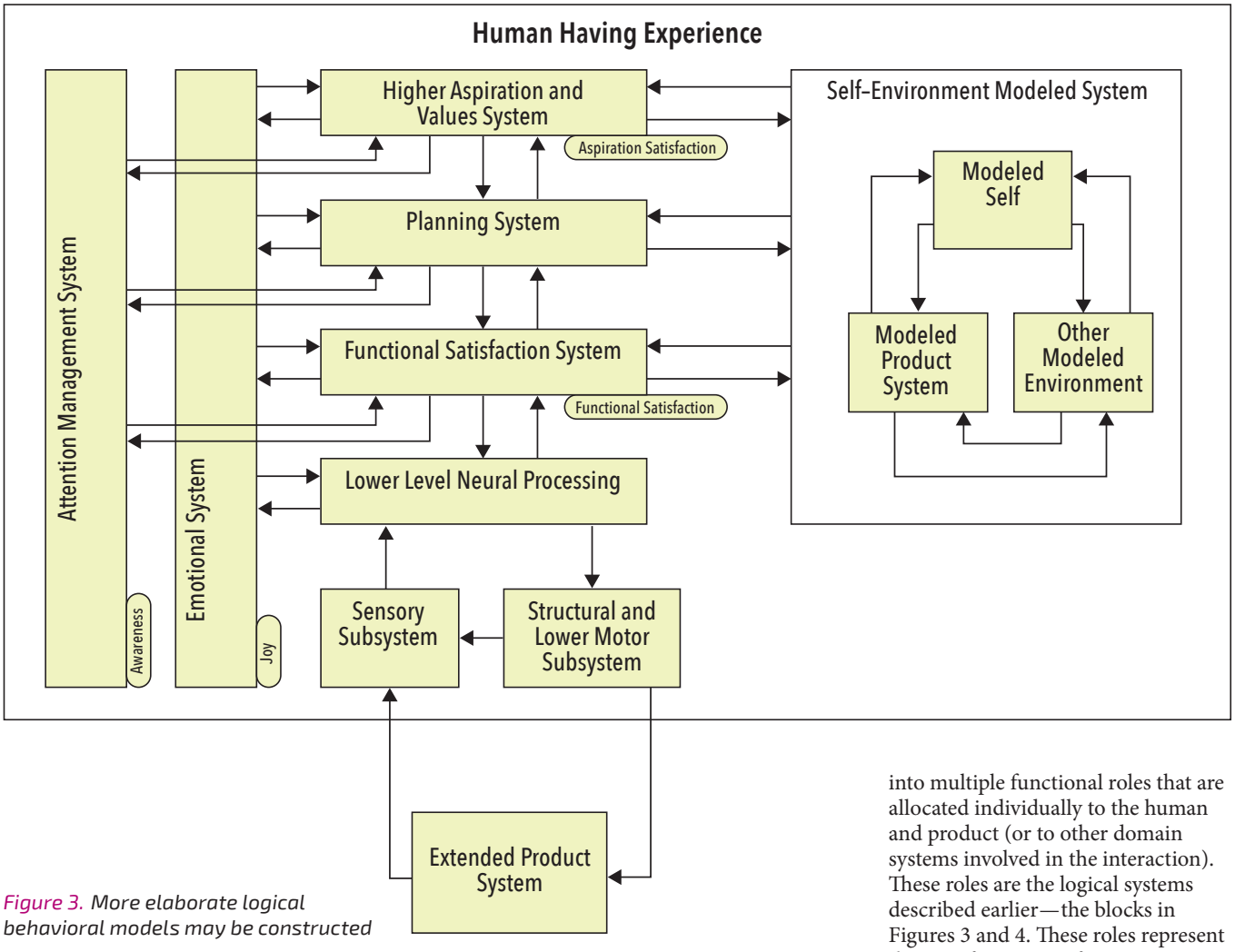


Figure 3. More elaborate logical behavioral models may be constructed

the attributes that parameterize these roles, discussed in the next section.

ATTRIBUTE COUPLINGS: MODELS OF INTER-DEPENDENCIES

Integrating the two parts of the model now enables more fully expressing how the structure of "soft" requirements on the product is dependent upon the structure of the human behavior model. For a human directly interacting with the product (see later herein for indirect cases), the chain of dependencies is as follows:

1. Feature and Feature Attributes: The "soft" requirements are imposed by the human experienced (subjective, psychological, emotional) outcomes we seek to optimize by the behavior of the product. The most "outcome oriented" of these characteristics important to product stakeholders will eventually be modeled as the features and feature attributes of the product—even though they describe human experienced outcome states. (See Figures 5 and 6.) This is because the product's attributes express how well it satisfies associated soft human requirements. All features and feature attributes are associated with feature stakeholders and are always described in the language of their stakeholders, not the technical requirements language of designers. Soft human requirements therefore are described in the language of the human stakeholder or specialists in human studies.
2. Functional Roles and Role Attributes: The features are associated with the functional interactions through which they are experienced. These are physical interactions of the human and product for the cases discussed here, during which physical input-outputs are exchanged between the human and product. These functional interactions are the systems engineering "glue" that ties together the human and product subsystems. These interactions are in turn broken into multiple functional roles that are allocated individually to the human and product (or to other domain systems involved in the interaction). These roles are the logical systems described earlier—the blocks in Figures 3 and 4. These roles represent the transformation of inputs into outputs, shown in those diagrams. The input-output transformations can be quantitatively described by prose statements, empirical graphs or tables from experience, by equations, by rules of thumb, results of focus groups or surveys, or other transformation descriptions, shown as requirements statements in Figure 5. These transformation descriptions are parameterized by attributes of the roles shown in Figures 3 and 4. These are "knobs" on the transformations that "tune" their input-output characteristics. The roles they parameterize serve to package, organize, and express the development team's best available (hopefully advancing) current knowledge, whether empirical or otherwise, as explicit intellectual assets (IP). *The coupling of feature to functional interaction to functional role spans and integrates the two worlds of soft human experienced qualities and hard technical requirements.*

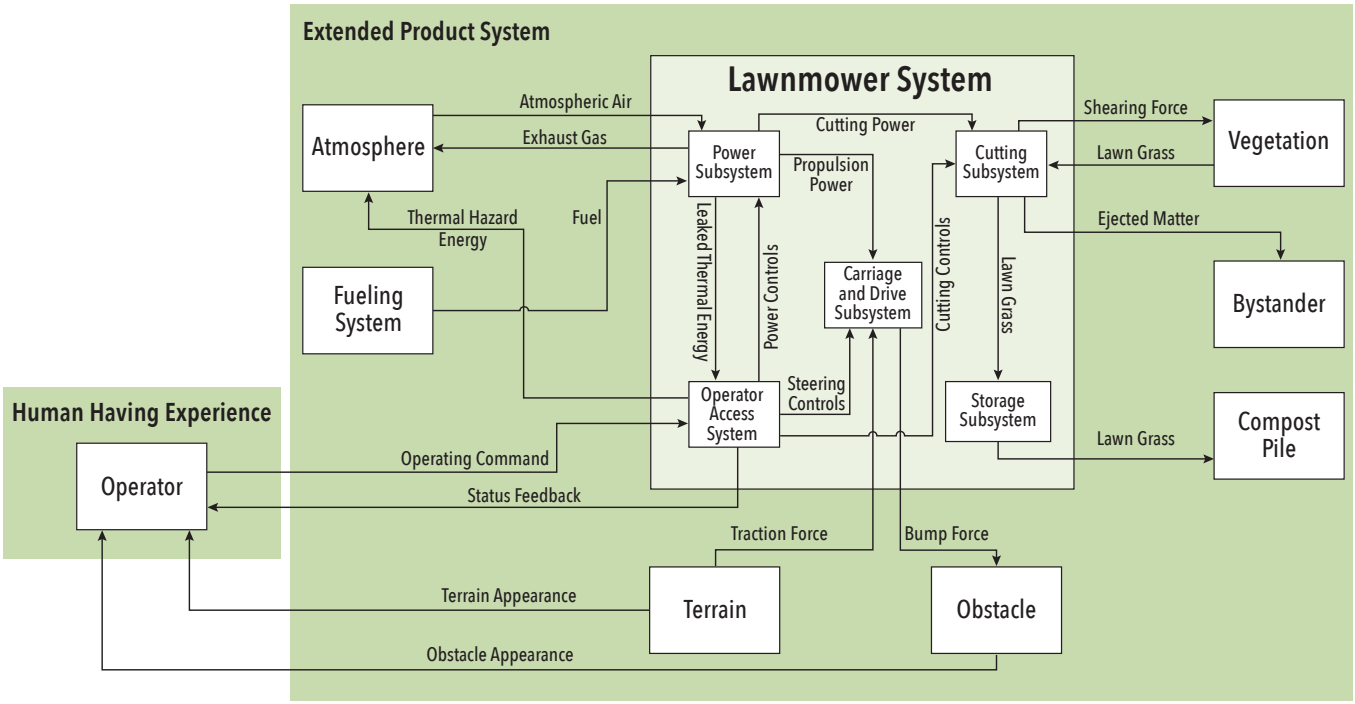


Figure 4. Product system domain and logical architecture

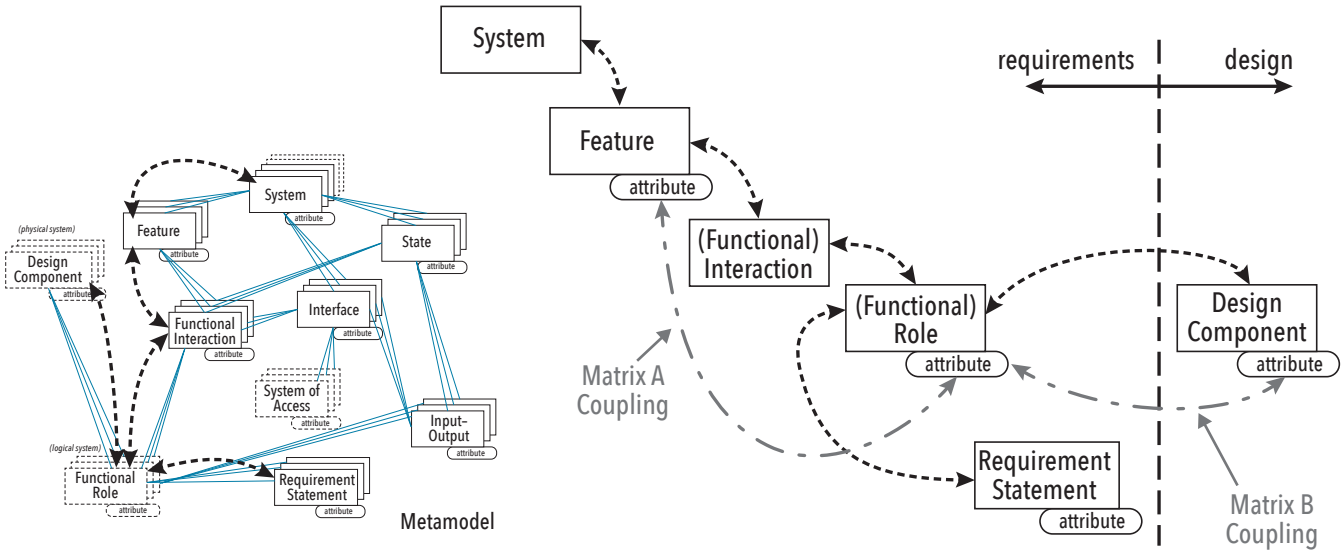


Figure 5. Tracing a subset of the metamodel

3. Design Components and Design Attributes: The architecture of the product design is expressed by the (physical) design components and their physical relationships, onto which are allocated the functional roles (behaviors). The design is further parameterized by the attributes of the (physical) design components, themselves tuned to best meet the role-based behavioral requirements.
4. Attribute Couplings: The dependencies of the three types of attributes shown in Figure 5 are expressed by attribute couplings, also summarized there. Design component attribute values are chosen to satisfy technical requirements expressed through functional role attribute values. These role attribute values are in turn chosen to satisfy feature attribute values that express stakeholder needs. These couplings express the dependency of hard technical requirements (as well as design) upon soft human experienced aspects. This also shows how to embed techniques such as QFD (Clausing et al. 1988) in the larger framework of model-based systems engineering. It explains the ideas behind the parametric requirements models supported by SysML (SysML Partners 2004).

A SIMPLIFIED EXAMPLE

This process ultimately leads to some quantitative expression of the organization's best known (whether empirical, analytical, rule of thumb, or other form of) knowledge about the coupling of subjective feature attribute outcomes to technical role attribute values. As a simple example, the "A Matrix" of Figure 7 (see also 5 and 6) expresses the organization's knowledge that a number of

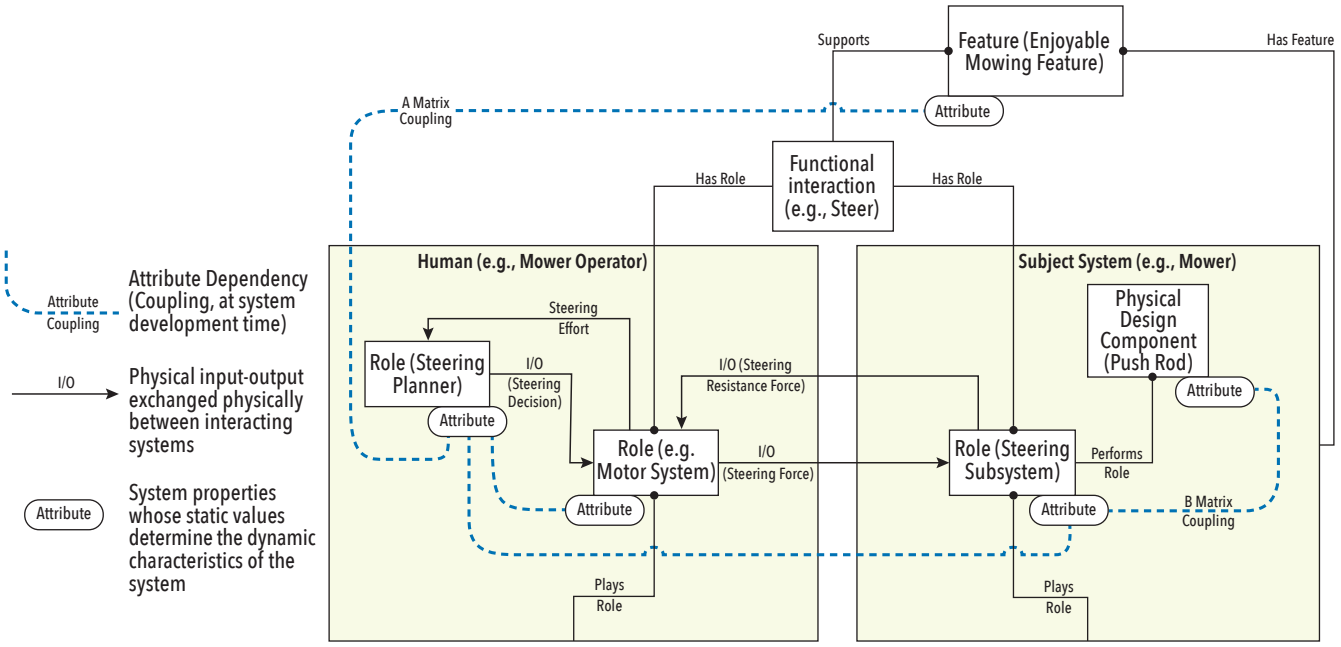


Figure 6. Attribute coupling framework

subjective human operator feature attributes for a lawnmower system are coupled to more technical role attributes describing that product's hard technical behaviour. The "X" indications in this matrix represent knowledge of couplings. This may include several levels of knowledge:

1. The simple (binary—yes or no) awareness that there is any significant coupling at all;
2. Awareness of the strength of the coupling;
3. More quantitative knowledge of the coupling (graphs, prose, simulations, tables, field surveys, rules of thumb, standards, etc.–in each case relating the coupled attributes).

In this example, a graph expresses knowledge of the relationship between the lawnmower operator's subjective sense of control (a "feeling") and the steering sensitivity and operating speed of the mower. Instead of a graph as shown, a table of values might have appeared. Still another possibility might have been a reference to a past study or to a person known by the organization to be the current expert on the subject. No matter what the form of the representation of the quantitative coupling relationship, the same framework can be used: couplings of attributes on stakeholder

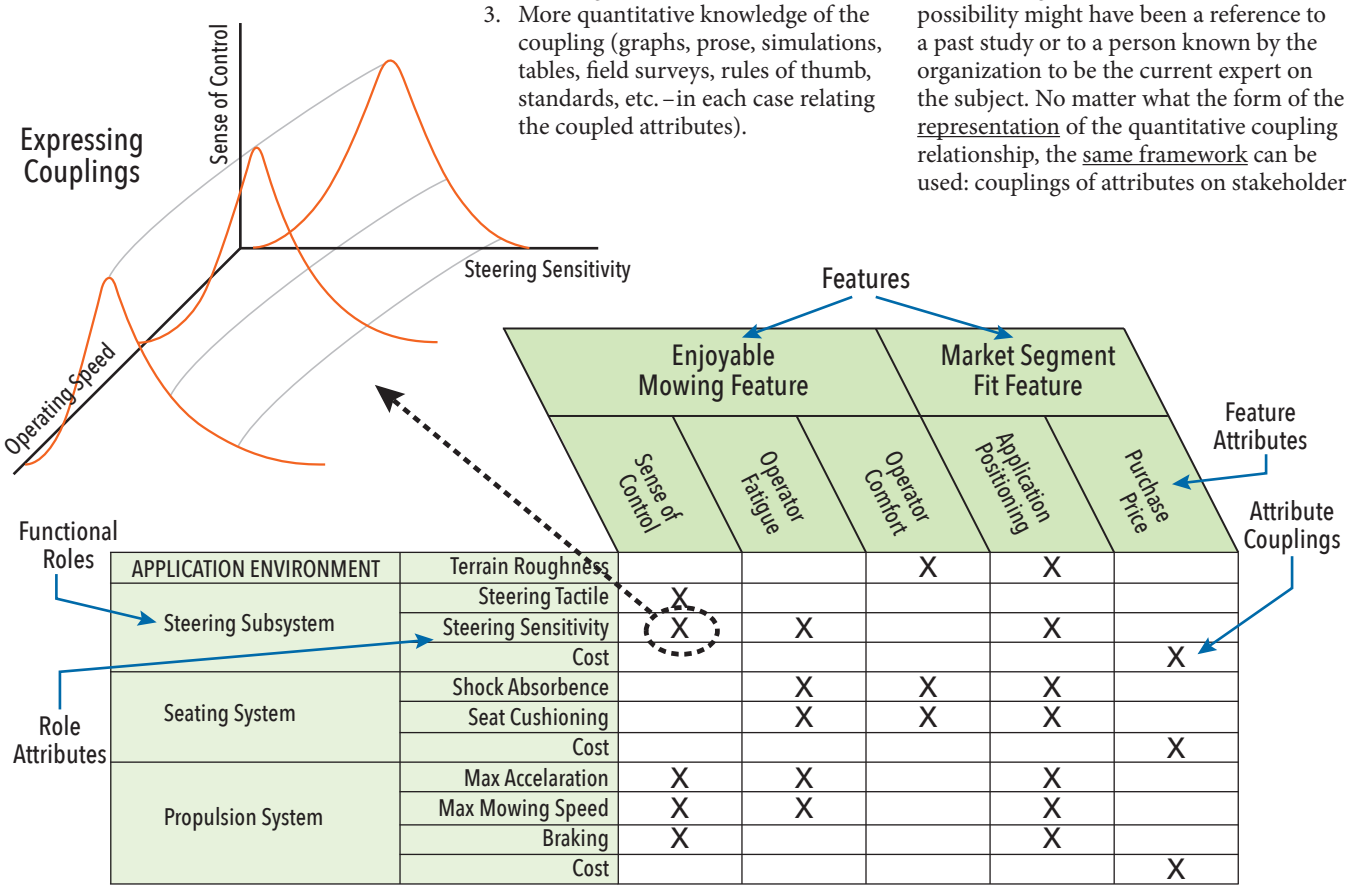


Figure 7. Simple example of attribute coupling Matrix A (features-roles)

features (including soft/subjective human experience outcomes) to technical role requirements attributes—a universal (and QFD-like) paradigm.

The actual prose form (input-outputs, attributes, relationships) of associated model-based requirements statements(s) is described in detail in (Schindel 2005)—the current paper shows how such prose requirements apply when models describe psychological or human factors, improving requirements effectiveness. As shown in that reference publication, requirements in this form are less ambiguous, easier to inspect for completeness, and easier to test, because they are embedded in and a part of explicit semantic models. This extends the use of prose-only glossaries to "explicate" the meaning of requirements statements using more descriptive models.

This approach also enhances validation and verification of human-oriented aspects. The validation of the feature-role couplings summarized by the A Matrix checks our understanding of human behaviour—not that of the product, but still essential to validating its requirements— and can frequently be addressed through simulation (or prototyping or other approach) of the product to humans. The verification of the logical role-design component couplings summarized by the B Matrix verifies that a product design meets the technical input-output (black-box) requirements. Finally, an overall validation of a real designed product in the hands of the human combines these two in an end-to-end test. Separating them improves understanding of both the stakeholders and the product.

EXTENSIONS: ALL SYSTEMS ARE SOFT

It turns out that the above techniques are important for designers of all systems — not just those who design direct human-interaction types of products.

All Engineered Systems Have Human Stakeholders. Many products and systems don't have direct interactions with humans while performing their primary mission, thereby seeming to avoid the human experienced qualities challenges described above. A submersible pump in a deep well, an orbiting surveillance system, an undersea communication cable, and other even less isolated systems may conduct their primary missions without direct human interactions (notwithstanding the parts of their life cycles involving direct human interaction for fabrication, installation, or maintenance). Many such products primarily interact with other hard technology systems, instead of people, in performing their primary mission. The engineer may believe that the requirements of such systems are easier to specify than those that directly interact with humans, and their designers may be envied by the designer who must deal with more human-intensive systems. Indeed, prose form technical requirements for these systems may be generated by physical interaction model-based means (Schindel 2005).

However, to claim that this avoids human factors challenges is to overlook a critical commercial fact of life. All engineered systems are created for some intended purpose, and on behalf of some human stakeholder, even if the stakeholder is not a direct user of the system. Stakeholders represent a form of market for the system to be designed. Stakeholders or their representatives may include purchasers, shareholders, financiers, general managers, sales organizations, customers of customers, regulators, and others. The "markets" they populate value those systems on a relative scale, ranking some products over others. The difference in these valuations can spell the difference between commercial success and failure in competitive markets. The engineering organizations of these businesses will eventually discover that the judgments rendered by such markets are themselves something other than the objective stuff of physics. The *perceived value* of a pump, satellite, or cable includes *subjective judgments* made by humans. The *relative utility* of these systems is the subject of utility theory (Bell et al. 1988, Keeney et al. 1993, Nash 1950, von Neumann et al. 1953) which is itself embedded in the study of human psychology and mathematics. Every engineered system, no matter how technical, is ultimately subject to "soft" human judgments, and these are overlooked at the peril of the designer.

All Engineered Systems Require "Life Cycle Marketing Support." Whether sold into commercial markets or defended to institutional administrators, every engineered system requires marketing support over its life cycle, including connecting its engineering process to the "marketplace" for that system. The marketplace description and advertising of commercial consumer and industrial product and service offerings have evolved in sophistication through over a century of modern practice. Today, products are subject to "positioning" by planners, to occupy certain "mental spaces" in the marketplace, with respect to perceptions of competition, the buyer's self-image, and other factors (Trout et al. 1981). Although one result of this positioning effort is the content of product advertising and promotional campaigns, we are frequently reminded that our actual engineered products need to back up (or drive!) the claims of advertising with real performance that is consistent with those claims. We need assurance that our promotional programs and product designs will reinforce each other for an optimum use of the assets employed. However, they are described in the languages of very different professions and organizations. As a result of this built-in disconnection of perspective, the disparity between "what engineering designed", "what marketing sold" and "what the customer wanted" has become the subject of popular cartoons. When product positioning promotes various images of a psychological nature using the power of suggestion and association, how does the product design engineer practically incorporate these "requirements" into the actual technical specification and design of the product? We again have a case of "soft versus hard" requirements.

All Product Stakeholders Eventually Interact at Least Indirectly with the Product. In systems engineering terms, it may seem a long way indeed from the physical aircraft to the aircraft company's shareholder, but their (indirect) physical interaction is very real and important.

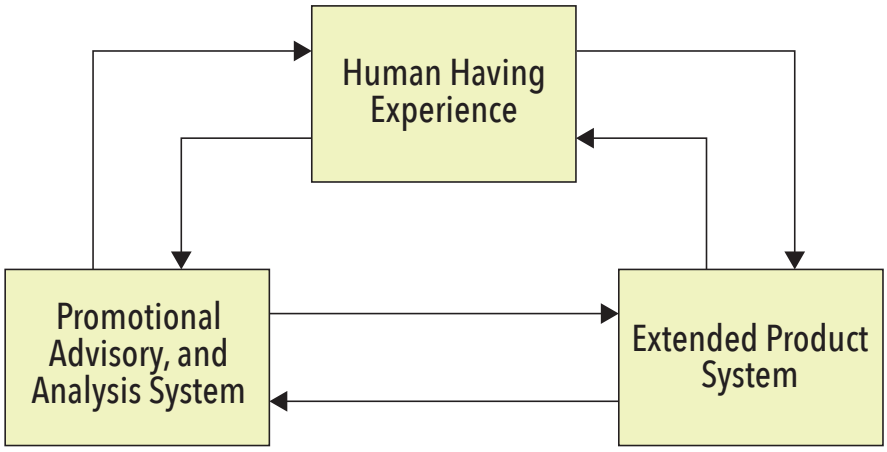


Figure 8. Extensions to the abstract model of Figure 1

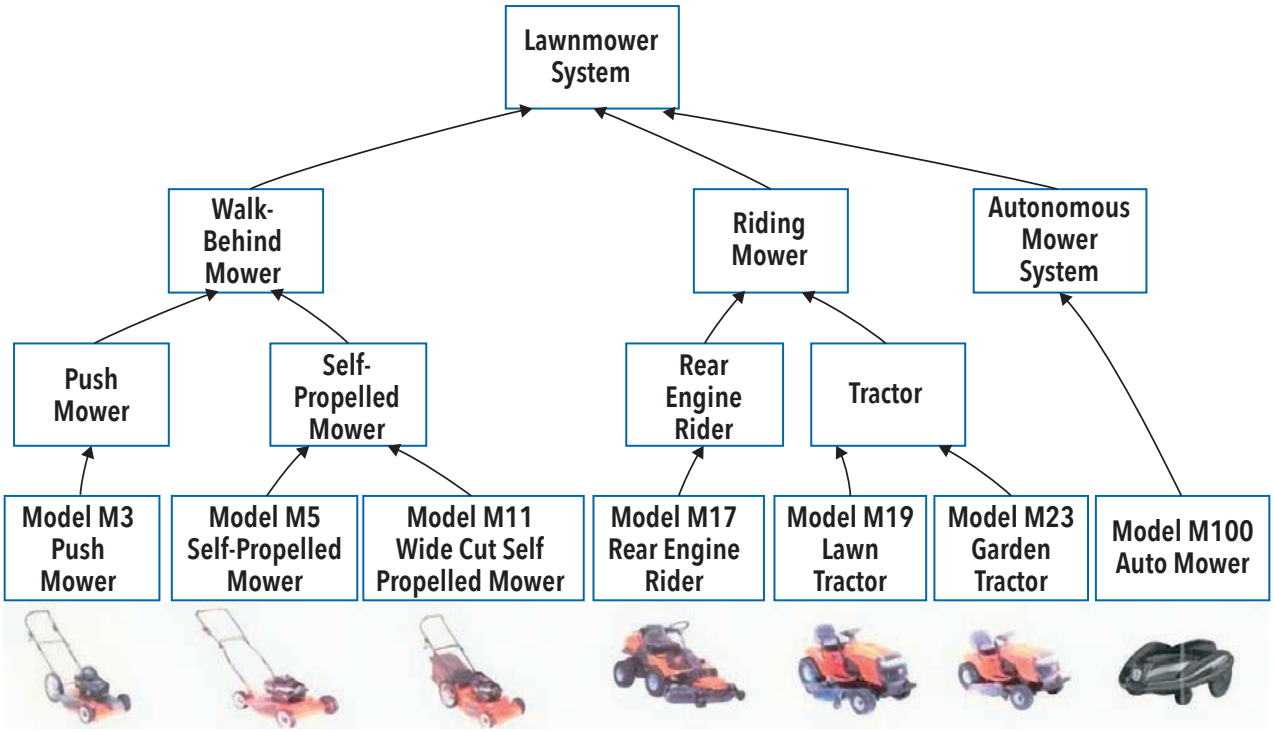


Figure 9. Patterns of soft and hard requirements, configurable across product lines

Without it, there would be no relationship whatsoever between the price of company shares and the performance of the aircraft. Investors will indeed debate how "real" this coupling is when they see peculiar share price performance in comparison to the product. But this equity market complexity only serves to underline the points of this paper concerning "soft" requirements. Were a stakeholder to be totally isolated from even indirect interaction with the product, then by definition they would have no stake in that product—a contradiction proving the point.

We can now return to Figure 1, add to it as Figure 8, and re-interpret it more generally, to extend the methods described in this paper:

1. Human Having Experience: This is a human being that interacts *even indirectly* (that is, *through intermediate people or other systems*) with the subject system within the extended product system, for whom we want certain experience-based outcomes.
2. Promotional, Advisory, and Analysis System: This is a system that communicates product-related usage information to and from the human user and product system (either one or both). This is intended by the supplier of the product system to (a) cause selection and purchase of the product system or the services it provides, (b) communicate advice on how to use or interact with the product system, and (c) collect and analyze information on how the user thinks about, selects, or interacts with the product system. (This "system" need not be high technology in nature—it could be based on mail telephone surveys, focus groups, consumer observation, or more sophisticated web sites or built-in monitoring technologies.)

All human stakeholders in the subject system are therefore included in this definition. Every such stakeholder is associated with features of the subject system that represent the value-centric outcomes that the stakeholder seeks from the subject system. These are subject to the same models and model coupling methods as described earlier above.

The informational "messages" produced by the promotional and advisory system and consumed by the human user are meant to establish the preliminary models of the self-environment modeled system even before experience with the product system. The physical interactions with the product system are required to reinforce those same models.

PATTERNS: LEVERAGING EXPERTISE ACROSS PRODUCT LINES

Understanding the soft requirements of human stakeholders and how they imply technical product requirements is highly valuable to a competitive organization, and not lightly accomplished. The resulting knowledge represents some of the most valuable intellectual assets of the organization. Preserving this information for repeated use across different configurations of products or systems in a large product line enterprise is highly desirable. The models described in this paper can be made to be configurable across product lines or system families, to meet differing market segment or application requirements. (Refer to Figure 9.) This leverages the knowledge of the most expert players and makes it available across the organization.

CONCLUSIONS AND RESULTS

Summarizing the results and conclusions:

1. Decomposed as described in this paper, "soft" requirements can be expressed in the form best-suited to the human experienced disciplines in which these arise (human factors, marketing, psychology, consumer research, cognitive science), but directly coupled to "hard" engineering requirements without loss of fidelity. This aids both forms, and unifies traditional disciplines for soft requirements with both technical requirements writing and model-based development.
2. The shared understanding of multi-disciplinary teams can be improved, by better understanding the origin of hard requirements in soft human

factors, and the form of their interdependent coupling.
3. Expressing couplings to other stakeholders, the same techniques can be used to express all stakeholder requirements, improving the understanding of stakeholder perspectives by the technical design team.
4. This improves the ability to write, understand, inspect, and use hard requirements, and improves the usual discipline of writing requirements statements, while maintaining traditional principles of requirements.
5. This approach also improves the ability to create requirements patterns—libraries of configurable, re-usable requirements, improving the performance of the engineering process across larger product line and COTS enterprises.
6. The treatment of soft requirements by methods such as QFD and axiomatic design can be unified with the total development process.
7. Automated modeling and requirements tools can increase in their capabilities using this paradigm. We have applied this approach using the systems engineering and modeling tools of a number of tools suppliers.
8. Less experienced engineers can apply these concepts to improve their requirements writing and modeling. We have successfully taught this approach to undergraduate and graduate engineering students, as well as practicing engineers in commercial and mil-aero organizations. ■

REFERENCES

- AP233 (ISO 10303). 2004. Web Site, http://step.jpl.nasa.gov/AP233/.
- Barberà, S., P. Hammond, and C. Seidl, eds. 2004. *Handbook of Utility Theory*. ISBN: 1-4020-7965-6.
- Bell, D. E., H. Raiffa, H., and A. Tversky, eds. 1988. *Decision Making: Descriptive, Normative, and Prescriptive Interactions*. New York, US-NY: Cambridge University Press.
- Clausing, D., and R. Hauser. 1988. "The House of Quality." *Harvard Business Review*, May/June.
- Crick, Francis J., 1994. *The Astonishing Hypothesis*. Charles Scribners.
- Damasio, Antonio R. 1994. *Descartes' Error: Emotion, Reason, and the Human Brain*. Putnam.
- De Laszlo, Violet S., ed. 1993. *The Basic Writings of C. G. Jung*. New York, US-NY: Random House.
- Edelman, Gerald M. 1989. *The Remembered Present: A Biological Theory of Consciousness*. Basic Books.
- Flinchum, Russell. 1997. *Henry Dreyfuss, Industrial Designer: The Man in the Brown Suit*. New York, US-NY: Rizzoli International Publications.
- Gaukroger, S. 1995. *Descartes: An Intellectual Biography*. Oxford, GB: Clarendon Press.
- Hutchins, Robert M., editor. 1952. *The Major Works of Sigmund Freud*. Great Books of the Western World, Vol. 54, William Benton, Publisher.
- INCOSE MBSE. 2004. INCOSE Model Driven System Design Working Group web site, http://www.incose.org/practice/techactivities/modelingtools/mdsdwg.aspx.
- James, William. 1950. *The Principles of Psychology*. Volumes 1-2. Dover Publications.
- Keeney, R. L., and H. Raiffa. 1993. *Decisions with Multiple Objectives: Preferences and Value Tradeoffs*. New York, US-NY: Cambridge University Press.
- LaBerge, David. 1995. *Attentional Processing: The Brain's Art of Mindfulness*. Cambridge, US-MA: Harvard University Press.
- Loewy, Raymond. 1998. *Industrial Design*. Overlook TP.
- Maslow, Abraham H. 1962. *Toward a Psychology of Being*. D. Van Nostrand Company.
- Nash Jr., John F. 1950. "The Bargaining Problem." *Econometrica* 18: 155.
- Pfeiffer, Bruce B., ed. 1993. *Frank Lloyd Wright: Collected Writings*. New York, US-NY: Rizzoli International Publications.
- Piaget, Jean. 1971. *Biology and Knowledge: An Essay on the Relations Between Organic Regulations and Cognitive Processes*. Chicago, US-IL: The University of Chicago Press.
- 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, November.
- Schindel, William D. 2005. "Requirements Statements Are Transfer Functions: An Insight from Model-Based Systems Engineering." INCOSE 2005 International Symposium, Rochester, US-NY, July.
- Sullivan, Louis H. 1956. *The Autobiography of an Idea*. Dover edition, New York, US-NY: Dover.
- Suh, Nam P. 2001. *Axiomatic Design: Advances and Applications* New York, US-NY: Oxford University Press.
- SysML Partners. 2004. Web Site, http://www.sysml.org/.
- Trout, J. and A. Ries. 1981. *Positioning: The Battle for Your Mind*. McGraw-Hill.
- von Neumann, J., and O. Morgenstern. 1953. *Theory of Games and Economic Behavior*. Princeton, US-NJ: Princeton University Press.
- Zadeh, L. A. 1965. "Fuzzy Sets." *Information and Control*, 8: 338-353.

ABOUT THE AUTHOR

William D. Schindel is president of ICTT, Inc., a systems engineering company, and developer of the Systematica™ methodology for model and pattern-based systems engineering. His 36 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, 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.

OCTOBER 2024
VOLUME 27/ ISSUE 5

Failure Analysis: Insights from Model-Based Systems Engineering

William D. Schindel, schindel@ictt.com

■ **ABSTRACT**
Processes for system failure analysis (for example, FMEA) are structured, well-documented, and supported by tools. Nevertheless, we hear complaints that FMEA work feels (1) too labor intensive to encourage engagement, (2) somewhat arbitrary in identifying issues, (3) overly sensitive to the skills and background of the performing team, and (4) not building enough confidence of fully identifying the risks of system failure. In fairness to experts in the process, perhaps such complaints come from those less experienced—but even so, we should care how to describe this process to encourage better technical and experience outcomes. This paper shows how model-based systems engineering (MBSE) answers these challenges by deeper and novel integration with requirements and design. Just as MBSE powered the requirements discovery process past its earlier, more subjective performance, so also can MBSE accelerate understanding and performance of failure risk analysis—as a discipline deeply connected within the systems engineering process.

WHAT WOULD WE LIKE TO IMPROVE UPON

Challenges of Traditional Failure Analysis Processes. Processes for system risk and failure identification, analysis, and planning are well-known, documented, and frequently supported by tools. These include failure modes and effects analysis—FMEA (Dyadem 2002, 2003; ISO/IEC 2006, 2007; US DoD 1980), fault tree analysis—FTA (Hyatt 2003), reliability centered maintenance planning—RCM (Moubray 1997), process hazards analysis—PHA (Hyatt 2003), and hazards and operability analysis—HAZOP (Hyatt 2003). Those who perform these sometimes voice challenges of these processes, such as the following:

1. Frequently labor intensive or tedious, adding cost and sometimes discouraging to the energy of those who face the next session;
2. May overlook certain problems, or feel somewhat arbitrary in identifying issues;
3. Typically, outcome is very sensitive to the skills and background of the performing team;
4. May not feel systematic in fully identifying the risks of system failure.

These lead us to ask: How can processes for failure identification and analysis be made to feel more systematic and less arbitrary and exhausting? How do we gain assurance we have found all the important failure modes and effects for a system? These and other challenges of traditional systems engineering approaches are being addressed using model-based systems engineering (MBSE).

ASSUMED MBSE BACKGROUND WE'LL NEED

The Emergence of Model-Based Methods. Model-based methods supplement the use of natural language prose in traditional engineering documents with the use of "models" which are explicit data structures (typically relational tables and formal diagrams). The structure of these models can be exploited to create analyses and checks that would be much more difficult and subjective to perform using purely prose-based methods. When applied well, they can also more effectively convey shared meaning to human readers. There is a growing literature on model-based systems engineering (MBSE) (Estafan 2009, Hybertson 2009, INCOSE 2009, Schindel 2005a). In this paper, we will focus on how failure analysis can be more deeply integrated as a part of such MBSE models.

Base MBSE Metamodel. The failure analysis approach this paper describes uses the fact that the requirements and high level design of a subject system can be represented in an information structure summarized by the base systems engineering metamodel of Figure 1.

Among the impacts of this metamodel is the re-positioning of prose functional requirements statements, which become a formal part of the model, as input-output relationships describing external system "black box" behavior during interactions with external actors—a kind of "prose transfer function". This is important to the results discussed in this paper and is described and illustrated in Schindel (2005a).

The failure analysis approach this paper describes also uses the fact that the (modeled) features for a system summarize, in stakeholder language, (all of) the behaviors of the system that will be valued by (all of) the system's stakeholders.

The balance of this paper assumes the availability of a systems requirements and

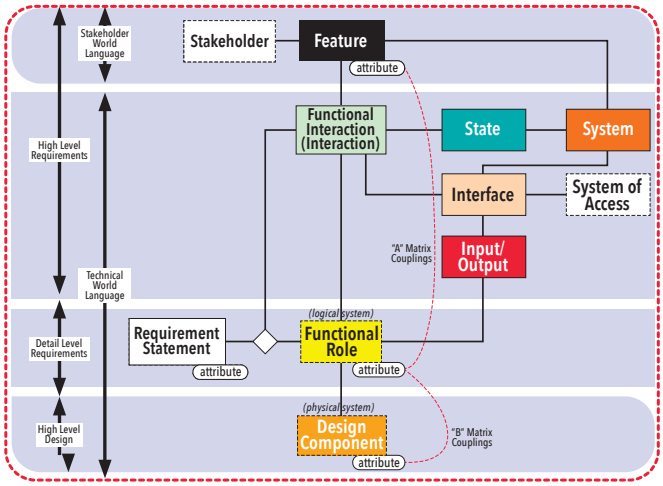


Figure 1. Summary of base systems engineering metamodel

design model that is based on the above metamodel. When we build on the foundation of the MBSE metamodel, some surprising, powerful, and unifying simplifications begin to appear.

MODEL-BASED FAILURE ANALYSIS: UNIFYING CONCEPTS

Features, Failures, and their Impacts. Let us assume that we have a modeled set of product requirements, based on the above metamodel. Because we have available all modeled system features satisfying all system stakeholders, it follows that a failure is then synonymous with not delivering what a feature promised. Because they are stakeholder ideas, modeled features are typically not very technical in their descriptions, but in fact summarize everything that a system should deliver to its stakeholders. (This may include stakeholder-quantified feature attributes.) Each feature is used to generate one or more failure impacts, summarizing the impact of not delivering (at least some aspect of) the feature's promise to the stakeholder. For example:

- Feature = "The system delivers medication on a dose accurate basis."
- Stakeholder impacts of not delivering Feature = "Illness," "Disability," "Death," etc.
- Severity of impacts: 3, 4, 5.

As illustrated above, each impact can also have a pre-populated severity of associated with it, describing the stakeholder-rated severity of such an impact ever occurring. Notice that this has been done so far without reference to the physical design of the system.

To cover all the stakeholders, features may include issues important not only to system end users, but also to those who manufacture, distribute, sell, or support the system, as well as shareholders in the profit-making enterprise, etc. We may or may not be interested in failure impacts on all these stakeholders and are offered the opportunity to explicitly decide. If a failure analysis is to be limited to certain stakeholder and feature subsets, such as medical harms to patients, then the only features that need to be considered are those that have those impacts on patients.

Surprise Number 1. Our first "surprise" is that *the only effects (the E in FMEA) that a failure can have are non-delivery of feature promises* — and these can be pre-modeled with each of the features, as failure impacts. If we claim to know our stakeholders and their modeled features, we can "pre-populate" the only possible effects of failures. If we think we have discovered an effect that is not implied by an existing modeled feature, we need to inform the feature modeler that they may have missed an important product feature. If we don't have a model of our system's stakeholders and their modeled features, the extended team has important homework to do before we can perform an FMEA or similar analysis. (This was always true in any method but is made more transparently obvious by the model-based approach.)

Requirements, Interactions, and Counter-Requirements. This approach also uses the fact that the (model-based) functional requirements statements for a system describe its required behavior, occurring during the interactions the subject system has with external systems (actors). Any failure of that system will include at least one instance of an interaction behavior by the system with at least one external system, having negative stakeholder consequence. At a black box level, these are the functional failures identified in FMEA, RCM, or other failure analyses. This method builds failure analysis on top of the system's requirements model, suggesting that the failure analysis cannot be completed without an agreed set of functional requirements, in model form. (Note that model-based requirements of the type described here are a technical characterization of relevant aspects of the system's black box behavior. This degree of "completeness" is characteristic of model-based requirements of the type discussed here. This "completeness" will now come in handy, for generating FMEA functional failures. This also makes it even more obvious why the system requirements as viewed by the requirements analyst, designer, and failure analysis review team should all be the same modeled requirements — and that each team can improve upon the shared model work of the others.)

Each system requirement statement is used to generate at least one counter-requirement statement. For example:

- Requirement = "The system shall deliver at least 3 hours of operation on one battery."
- Counter-requirement = "The system does not deliver at least 3 hours of operation on one battery."

A complete set of counter-requirements can be rapidly generated in a simple way from the system's requirements, by "reversing" them.

Surprise Number 2. *All FMEA functional failures can be rapidly generated as counter — requirements, from MBSE modeled functional requirements.*

Some requirements may generate more than one counter-requirement. For example:

- Requirement = "The system shall maintain temperature in the range 70-74 degrees."
- Counter-requirement 1 = "The system allows temperature to exceed 74 degrees."
- Counter-requirement 2 = "The system allows temperature to fall below 70 degrees."
- Furthermore, because the requirements were already associated with the features of a system model, the counter-requirements can be easily associated with impacts, which are the (feature non-delivery) "effects" of an FMEA analysis, without "from scratch" analysis.

Surprise Number 3: *All associations (match-ups) of FMEA functional failures with FMEA effects can be generated from the association of the violated requirements with its associated stakeholder feature.*

Modes (States): Failure Modes. The MBSE requirements approach referenced also uses the fact that the interactions a system has with external systems can be thought of as associated with the system being in a certain state, or mode. The behavior (external interaction) of a system is different if it is "off", "on", "idling", etc. Each of these are states (or modes) of that system's behavior. These are all "normal" modes, in the sense that while they occur in different circumstances, the associated system

behavior is considered normal (that is, what is described by requirements).

In addition, a system can sometimes enter an "abnormal" mode, in which its behavior is undesirable — such as "overheated". Sometimes abnormal states are called failure modes when the associated behavior is bad enough.

Interaction-State Chains; Causes. This approach further uses the fact that the design components, states, interactions, requirements, and features information of the Figure 1 metamodel can be unfolded (split) across normal and abnormal behavior, and across "causality chain" sequences. The resulting models add further to the information used to populate a failure analysis (for example, FMEA table).

In all these cases, the current mode (state) of the system can be viewed as the immediate reason that it is behaving a particular way. That behavior is characterized by the interactions the system is currently able to perform (the interactions associated with that state).

If we then ask how the system came to be in its current state, we find that a previous interaction of some sort will have "placed it in the current state". This leads to the idea that there are "causality chains" that take the form of sequences of alternating interaction, state, interaction, state, etc. For example:

- Interaction: Turn On the System
- State: System On
- Interaction: Request System Menu
- State: Displaying Menu.

This same idea works for abnormal states:

- Interaction: Insert Battery
- State: Battery Inserted Backwards
- Interaction: Turn On System
- State: System Inoperative.

In all these cases, the idea of cause can be pursued by looking to earlier parts of the chain. We can say that a later part of the chain is "caused" by the states and interactions of an earlier part of the chain.

Pre-Populating A Library of Failure Modes. The counter-requirements and feature failure impacts described earlier above depend only upon the structure of requirements and stakeholder expectations for a system — they are independent of its design. In contrast, the failure modes of a system depend upon its design—specifically, upon its physical design components. Each such design component has an expected behavior, based upon the logical roles and requirements allocated to it, and a set of failure modes, which are abnormal states that physical component type may enter in which it will display behavior violating its allocated logical roles and requirements.

Since counter requirements and feature failure impacts can be pre-populated independent of design, is it possible that failure modes can be pre-populated independent of system requirements? This turns out to be connected to knowing what roles and (decomposed, or white box) requirements will be allocated to the physical part. For most physical parts playing typical or "standard" roles, it turns out that we have such a prediction available even if the (parent black box) requirements of the total system are not currently visible. For example:

- Design Component = Madsen Model P53 Centrifugal Pump
- Normal Allocated Roles = Liquid Transport, Liquid Containment, Powered Safe Operation
- Failure Modes = Bearing Failure, Leakage Seal Failure, Short to Case
- Probabilities of Occurrence = 0.002, 0.00045, 0.000001 (per 10,000 service hours).

Probability of Occurrence. As illustrated above, for each pre-populated failure mode, we can also include probability of occurrence parametric information that characterizes the likelihood of the physical component entering the failure mode from the interactions it will experience in its typically assigned roles. This will later help to drive the failure risk scoring process in the usual manner.

Combinatorial Matching Up of Requirements and Design Data. The functional failures (counter requirements) and failure effects (feature failure impact) data can be pre-populated independent of the system's internal design, and the failure mode data for standard component roles can be pre-populated independent of the system's external requirements. So, when both the requirements and a candidate design have become known, how do these two halves of the failure analysis model get connected to each other? This turns out to be a combinatorial algorithm.

First, it turns out that the counter-requirements (functional failures) obtained by reversing the requirements statements may describe some hypothetical external behaviors that are never (or with probability too small to matter) caused by component failure modes. This will cause some pre-populated functional failures to be dropped. For example, a requirement that a product weigh less than one pound has a counter-requirement that it weighs more than one pound. It may be determined that there is no component failure mode that impacts weight, so that this functional failure is dropped from the list. (Notice that even this failure mode could happen for some products — for example, a hazard protection suit that becomes wet weighs more.)

Second, it turns out that some failure modes of a physical component have no consequence on the product's required behavior, because the failure mode describes a role not allocated to the part in this particular product design. For example, an integrated circuit may have built-in circuitry for performing certain functions which are not used by a certain product's design, even though other portions of that chip are used.

The connection of the requirements half of the failure analysis to the design half of the failure analysis is made by matching up "mating" pairs and discarding what is left as not applicable (after checking for missed cases this approach also helps us find—another benefit). The matching up is accomplished through the matching of counter-requirements with failure modes. Each failure mode causes some abnormal behavior. All abnormal behavior is described by counter requirements. When we find a counter-requirement belonging to a failure impact is equal to a counter-requirement for a failure mode, that pair is associated together, completing two major sections of a row in a failure analysis table. (Some failure modes may connect to multiple counter requirements and some counter requirements may connect to multiple failure modes.)

This process may use two levels of requirements, in the form of system black box requirements and their decomposed white box requirements (allocated to physical parts), in which case counter-requirements may be developed at both levels. A simpler alternate method is to use only one level of counter-requirements, with the component failure modes associated directly with the resulting abnormal behavior at the black box level—in which case the association of failure modes with abnormal behavior is dependent upon knowing the system level design. Likewise, the states discussed above may be at two levels, representing states (and failure modes) of system components and the whole system, or simplified to states of the whole system, in which case the failure modes are modes of the whole system and again dependent upon its design.

The discussion above assumes failure modes originate in internal system components, typical of analyses such as a design FMEA (D-FMEA). Also discussed later below are failure modes of external people or processes that impact upon the subject system, as seen in an application FMEA (A-FMEA) or a process FMEA (P-FMEA).

The counter-requirements matching-up approach is substantially the same in these cases.

A UNIFYING MBSE VIEWPOINT FOR RISK ANALYSIS INFORMATION

Order of Occurrence versus Order of Analysis; Checking; FMEA versus Fault Tree. FMEA analysis typically reasons from component failure modes to system level counter-requirements, to the stakeholder impacts (failure effects, such as user injury). This traditional analysis thus occurs in the sequence of cause-to-effect, and the methodology described here supports that order of reasoning. In a traditional FMEA table, it proceeds more or less from left to right. This traditional order of reasoning is why FMEA is said to work for analysis of single failure modes but not multiple simultaneous failure modes.

This methodology also supports the generation of fault tree analyses. Whereas an FMEA analysis traditionally begins from each possible component level failure mode and reasons to its effect (typically a one-to-one process generating a row of an FMEA table), a fault tree analysis traditionally begins with each effect and reasons backwards to identify each possible component failure mode that might cause it (typically a one-to-many process generating a many-branched fault tree under a single effect). Each path of the fault tree is roughly equivalent to a row of the FMEA table. The information models described here describe both approaches, differing only by the order in which the data model is filled in during the analysis process. The use of MBSE failure analysis allows reasoning in other directions—because it is really about an underlying information model, not an order of reasoning, we can populate that information model in different orders. These include backwards reasoning from failure effect to cause (as in a fault tree analysis) and middle-out reasoning, from system counter requirement to both their upstream causes and downstream effects. This is of major value, as it facilitates completeness checking of the resulting failure analysis table. We can independently check the effects against a complete library of all possible feature-based impacts. We can independently check the middle (the system counter-requirements) against a complete library of all possibilities, based on the listed system requirements. This improves completeness and coherence of the FMEA or other analysis, including its inspectability.

Faults versus Failures; Fault Tolerant Systems; Fail Safe Aspects. In the specific language (Anderson and Lee 1981) of fault tolerant systems (which is not always used the same in failure analysis procedures) faults and failures are undesirable states or behaviors, but don't mean the same thing. A fault is an abnormal component or subsystem condition (state), which may or may not result in a system level failure. Remembering from above that failures are not delivering agreed upon stakeholder features, we can say that a fault tolerant system is a system that does not fail (continues to deliver features) in spite of component or subsystem faults. (That is, it tolerates faults in its own components, while continuing to deliver external features.)

For example, aircraft hydraulic systems typically employ redundancy, so that they can deliver safe flight services while tolerating a fault in a hydraulic line.

In the language of failure mode analysis, the term "failure mode" is frequently used to describe an abnormal state of a component or subsystem, even if the overall system was designed to keep delivering all its external services in the presence of that component failure mode. This is not so inconsistent if you consider that the subsystem or component is not delivering its "external" services, but it can be a little confusing if you don't expect the term or keep track of system levels. Sometimes a system internal fault can present risk of a serious (for example, life or property threatening) failure behavior by the subject system. In those cases, mitigations are sometimes planned such that, although the system may fail to deliver all its promised features, it protects from presenting a more serious failure. That is, it still fails, but "fails safely." This is called a fail-safe system.

Subsystem Causing Failure: D-FMEA. In a system, an abnormal state of a component may cause a system level failure. We can reason forward from the component state to the system failure it causes, or backward from the component state to its cause. For example, the following failure mode is "caused" by the interaction shown:

- (Interaction) Cause of Failure Mode: Normal Wear
- Component Failure Mode State: Gear Train Binding/Lash-Up.

Remembering the idea of interaction-state chains, we can see that many such failure mode states can be said to be caused by a previous interaction, whether it is a normal use interaction or some extraordinary damaging interaction. If the causal interactions are "normal" behavior by the external systems performing them, then we could say that the failure mode is effectively inherent to the design of the subject system in its normal use. Analyzing failures of this kind is typically the subject of D-FMEA (design failure mode effects analysis) work. Sometimes this leads to a different design to reduce the likelihood of the failure mode occurring, or in other cases to other controls (mitigations) intended to reduce the impact of the failure mode when it occurs.

In all those cases, it could be said that the role played by the subject system in normal interactions eventually leads to the failure mode of the system's component. However, it is alternatively possible that the system design is not the cause, but rather that the external systems are behaving abnormally. This case is covered in the next two sections.

Peer System Causing Failure: A-FMEA. External systems interacting with the subject system are sometimes called "peer" systems, or "actors". Unlike the subsystems or components discussed above, they are external to the subject system.

In an A-FMEA (application failure mode effects analysis), attention is focused on the effect of abnormal behavior by external systems that are typically human "users" of the subject system. It could be said that the original failure modes in this case are states of the external system. For example:

1	Failure Mode (Pilot State):	Attention Overloaded
2	Interaction:	Select Target (assume wrong value entered)
3	State (of Weapons System):	Awaiting Weapon Release Confirmation
4	Interaction:	Confirm Weapon Release
5	State:	Delivering Weapon

As illustrated by the above example, we can have a failure to deliver overall system features even though the subject system meets all of the requirements assigned to it. However, it is also possible for an external system to drive the subject system into its own abnormal (for example, damaged) state, after which it no longer meets requirements assigned to it. For example:

1	Cause of Failure (Interaction):	Poor User Training
2	Resulting Failure Mode (State):	User Unaware
3	Interaction:	User Closes Valve (Over-Tightening)
4	Resulting System Component State:	Valve Seal Failure

Both of these cases are of interest in an A-FMEA. The second case looks a lot like a D-FMEA after the point of driving the subject system into a bad state.

Notice that "users" are not the only external systems whose failure modes can damage the subject system's state. Other faulty systems in the application domain may also have to be considered. When the external actor that is in an abnormal state is a human being, the MBSE model is in the territory of modeling human behavior. This is further discussed in Schindel (2006).

Peer System Causing Failure: P-FMEA. One special external system traditionally analyzed is the subject system's manufacturing system. This is the subject of a P-FMEA (process failure mode effects analysis). The nature of a manufacturing system is to create the subject system, so it may be found that all the P-FMEA failures of interest result in bad product system states. For example:

1	(Interaction) Cause of Failure Mode:	Glue Build-Up on Nozzle During Use
2	Component Failure Mode State:	Nozzle Obstructed
3	(Interaction)	Not enough glue applied
4	Subject System State	Part Loose

There can also be manufacturing process failures that fail in the sense of not delivering on all the other manufacturing process systems features, as when manufacturing yield, manufacturing operating cost, or manufacturing safety are impacted by manufacturing faults. Depending on the intended scope of the P-FMEA, these may or may not be of interest to include and analyze.

Other major processes, such as the commercial distribution process, can have faults that create bad states in the subject system. For example:

1	(Interaction) Cause of Failure Mode:	Transport Packaged Product
2	Component Failure Mode State:	Package Seal Fractured
3	(Interaction)	Tolerate Exposure to Contaminants
4	Component Failure Mode State:	Food Product Contaminated

Depending on the intended scope of the P-FMEA, these other processes may also be considered.

D-FMEA, A-FMEA, P-FMEA, and Unified FMEA. Although it may be desirable to separate the D-FMEA, P-FMEA, and A-FMEA "reports" for attention by different groups, and to generate and review them using different subject matter experts, it is also desirable to generate them from a consistent underlying information model. For example, all three FMEA types depend on the same system level counter-requirements and feature impacts. If this consistency is used, then it is easier to understand the different FMEAs in a consistent way, and to judge their accuracy and completeness.

While there may be reasons to differently format or label the tabular "reports" that are generated for these different types of failure analysis, the approach described here at least intends to generate them from a common base of underlying information, and to minimize differences in labeling except where it improves the outcome.

FURTHER LEVERAGING THE RESULTS

Patterns As Re-Usable Models. This paper describes the use of model-based systems engineering information in failure analysis, to improve results. If an enterprise needs to perform failure analysis on different products or systems that are somewhat related but vary in their specific configuration (for example, product lines), then a more powerful extension is also available. This is called pattern-based systems engineering (PBSE). The basic idea is to make the models configurable and re-usable, so that they can rapidly be re-used in future projects and can also be used to accumulate learning. This is a bigger idea than accumulating standard lists of failure modes. See Figure 2.

This approach to systems engineering patterns treats a pattern as a configurable, re-usable model of requirements and design, described further in Schindel (2005b), and Schindel and Smith (2002).

Enhanced Use of FMEA and Risk Analysis Tools. A number of basic and more advanced commercial automated tools are available for use in generating FMEA and other forms of failure or risk analysis. In their most basic use, the analyst manually enters data into relatively fixed forms and generates resulting reports. In their more advanced form, these tools support customization or configuration of reports, data entry, and some aspects of the underlying information models. Some also support accumulation and use of re-usable standard categories or other data, and some support integration with other engineering tools, such as requirements management tools.

The model-based concepts, methodology, and procedures described in this document can be used with a number of these commercial tools, improving their value. In general, the more powerful and flexible the tool, the more aspects of this methodology may be used.

The simplest, but least beneficial, way to initially do this is to configure the tables and reports of a tool to accept manual entry of data of the type described in this document.

A more sophisticated approach allows re-use of data from a pattern of requirements, design, and failures (patterns). Since patterns are relational models, this is more powerful than simply having lists of standard pull-down items.

This methodology also enhances the ability to integrate an FMEA or failure analysis tool with a requirements management tool, by using counter-requirements that are associated with the system level requirements. This is more powerful than simply having links between data items in two tools. In fact, if a requirements and design model is available in MBSE form, then tool-based combinatorial algorithms can be used to automatically generate an initial draft FMEA table. Of course,

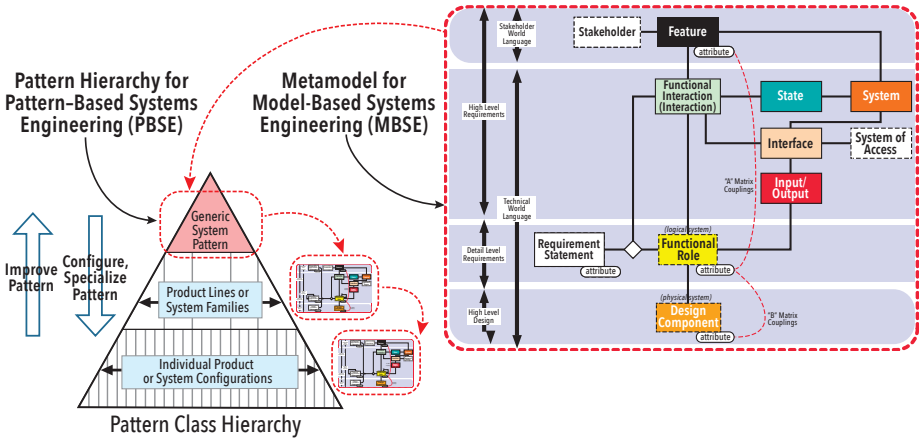


Figure 2. Patterns are re-usable, configurable models

this does not replace human analysis, but does reduce the drudgery of initial generation, freeing the analyst to do deeper thinking and analysis of the failure data.

RESULTS TO DATE

We have seen these methods help both experienced FMEA analysts as well as newcomers to more productively generate well-organized failure analyses, in applications including manufacturing and health care. The approach is not at odds with traditional methods, in producing substantially the same form of deliverable—but provides a stronger basis for understanding the meaning and degree of coverage that deliverable represents, while more tightly integrating failure analysis with requirements and design data.

CONCLUSIONS

1. Failure analysis data and processes can be more deeply integrated with system requirements data and processes, using model-based methods, with benefits to depth of shared team understanding, productivity, process cohesion, coverage, and lower level of entry expertise for participants.
2. A subset of FMEA analysis can occur in advance of, or independent of, system design, using the structure of model-based stakeholder features and functional requirements to pre-populate the space of potential functional failures and their prioritized effects.
3. Another major subset of failure analysis data can be pre-populated that is requirements independent, in the form of libraries of physical components (or technologies), their typically assigned roles, and their failure modes and associated abnormal behaviors.
4. Modeled system design introduces failure mechanisms for D-FMEA, while human, process, and equipment actors introduce failure sources for A-FMEA and P-FMEA, all of which can be better integrated.
5. FMEA, fault tree, and other forms of analysis can be viewed as different views of the same underlying modeled data, for different purposes and emphases.
6. Patterns, when formed as re-usable, configurable models of system requirements and design, can include failure risk analysis, whose coverage and quality can be improved from project to project, in support of a learning organization.
7. Automated tools for failure analysis, requirements management, design, simulation, and other aspects of the systems engineering process can be integrated more deeply than simply linking their data records, by configuring their databases to take advantages of the integrated underlying MBSE/PBSE metamodel. ■

REFERENCES

- Anderson, T., and P. Lee. 1981. *Fault Tolerance: Principles and Practice*. Prentice-Hall ISBN-10: 0133082547.
- Dyadem. 2002. *Guidelines for Failure Mode and Effects Analysis, for Automotive, Aerospace and General Manufacturing Industries*. CRC Press. ISBN 0-9731054-1-0.
- ———. 2003. *Guidelines for Failure Modes & Effects Analysis for Medical Devices*, ISBN 0849319102, Richmond Hill, Ontario, CA: Dyadem Press.
- Estefan, J. 2009. "Survey of model-based systems engineering (MBSE) methodologies." INCOSE-TD-2007-003-01, Rev B.
- Hyatt, N. 2003. *Guidelines for Process Hazards Analysis, Hazards Identification & Risk Analysis*. ISBN 0849319099, Richmond Hill, Ontario, CA: Dyadem Press.
- Hybertson, D. 2009. *Model-Oriented Systems Engineering Science*. CRC Press.
- INCOSE. 2009. INCOSE Model driven system design working group web site, http://www.incose.org/practice/techactivities/wg/mdsd/.
- ISO/IEC. 2006. IEC International Standard 60812 Analysis Techniques for System Reliability – Procedure for Failure Mode and Effects Analysis (FMEA). Geneva, CH.
- ———. 2007. ISO/IEC 14971:2007 Medical Devices—Application of Risk Management to Medical Devices. Geneva, CH.
- Moubray, J. 1997. *Reliability-Centered Maintenance*, Second Edition. New York, US-NY: Industrial Press, Inc.
- Powell, M. 2005. "Dealing with Uncertainty in Systems Engineering." Tutorial INCOSE 2005 International Symposium, Rochester, US-NY, 10-15 July.
- Schindel, W. 2005a. "Requirements Statements are Transfer Functions: An Insight from Model-Based Systems Engineering." INCOSE 2005 Symposium, Rochester, US-NY, 10-15 July.
- ———. 2005b. "Pattern-Based Systems Engineering: An Extension of Model-Based SE." Tutorial INCOSE 2005 International Symposium, Rochester, US-NY, 10-15 July.
- ———. 2006. "Feelings and Physics: Emotional, Psychological, and Other Soft Human Requirements, by Model-Based Systems Engineering." INCOSE 2006 International Symposium, Orlando, US-FL, 9-13 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, November.
- US DoD 1980. MIL-STD-1629A, "Military Standard Procedures for Performing a Failure Modes, Effects, and Criticality Analysis."

ABOUT THE AUTHOR

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.

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.

35th Annual INCOSE international symposium

hybrid event

Ottawa, Canada
July 26–31, 2025

Mark your calendar now!

26 – 31 July 2025

Submission date for papers, tutorials, panels, and paperless presentations: **30 November 2024**

Call for Papers

Call for Panels/ Roundtables

Call for Tutorials

Call for paperless presentations

Visit **www.incose.org/symp2025** and contact us **TODAY** - The INCOSE Events Team

CATIA MAGIC

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

LEARN MORE

DASSAULT SYSTEMES