PROJECT PERFORMANCE INTERNATIONAL

CERTIFICATION TRAINING INTERNATIONAL
A PPI Company

SYSTEMS ENGINEERING NEWSLETTER

PPI SyEN 69 – September 24, 2018

brought to you by

Project Performance International (PPI)

systems engineering training for project success

PPI SyEN is an independent free newsletter containing informative reading for the technical project professional, with scores of news and other items summarizing developments in the field, including related industry, month by month. This newsletter and a newsletter archive are also available at www.ppi-int.com.

Systems engineering can be thought of as the problem-independent, solution technology-independent, life-cycle-oriented principles and methods, based on systems thinking, for defining, performing, and controlling the engineering effort within a technical project. The approach aims to maximize the benefit delivered to the enterprise, as influenced by the needs and values of the other applicable stakeholders.

If you are presently receiving this newsletter from an associate, you may wish to receive the newsletter directly in future by signing up for this free service of PPI, using the form at www.ppi-int.com. If you do not wish to receive future Systems Engineering Newsletters, please unsubscribe by clicking on the link at the bottom of this email.

We hope that you find this newsletter to be informative and useful. Please tell us what you think. Email us at syen@ppi-int.info.

The views expressed in externally authored articles are those of the author(s), and not necessarily those of PPI or its professional staff.

IN THIS EDITION

5. Featured Organizations

5.1 Centre for Science and Policy University of Cambridge

5.2 International Society for System Sciences

5.3 ARC Advisory Group

5.4 SysAdmin, Audit, Network, and Security (SANS) Institute

5.5 Defense Advanced Research Projects Agency (DARPA)

Read More…

6. News on Software Tools Supporting Systems Engineering

6.1 AgileCraft: Scaled Agile Management Platform

Read More…

7. Systems Engineering Publications

7.1 Modeling and Simulation Support for System of Systems Engineering Applications

7.2 Deep Shift: Technology Tipping Points and Societal Impact

7.3 Notes on the Synthesis of Form

7.4 The ‘How’ of Transformation

7.5 If I Knew I was going to be a PM, I Would Have Followed a Different Path

7.6 Why Do Projects Fail?

7.7 How to Save a Failing Project: Chaos to Control

7.8 Partnering in Construction: A Practical Guide to Project Success

Read More…

8. Education and Academia

8.1 Systems Engineering at Cranfield University, UK

8.2 An Engineering Student’s Guide to Effective Note Taking

8.3 Applications for a Visiting Associate Professor Position in Software/Systems Engineering at Carlos III University of Madrid, Spain

Read More…

9. Some Systems Engineering-Relevant Websites

Read More…

10. Standards and Guides

10.1 ISO/IEC/IEEE 21839 - Systems and software engineering — System of Systems (SoS) Considerations in Life Cycle Stages of a System

10.2 Cybersecurity Requirements for Industrial Control Systems Development

Read More…

11. A Definition to Close On

System of Systems (SoS)

Read More…

12. Conferences and Meetings

Read More…

13. PPI and CTI News

Read More…

14. PPI and CTI Events

Read More…

15. PPI Upcoming Participation in Professional Conferences

Read More…

1. QUOTATIONS TO OPEN ON

“Effective risk management requires an understanding of the ingredients of risk: value, threat, vulnerability, and probability. Value includes the value of avoidance of a detrimental outcome, as well as the value of achievement of a beneficial outcome.”

Robert Halligan

“Nothing is more difficult and, therefore, more precious, than to be able to decide.”

Napoleon Bonaparte
French General and Emperor, 1769-1821

2. FEATURE ARTICLE

2.1 Systems Engineering Transformation

by

Troy A. Peterson

INCOSE, Assistant Director Transformation

SSI, Vice President

Email: tpeterson@systemxi.com

Abstract

While complex systems transform the landscape, the Systems Engineering discipline is also experiencing a transformation to a model-based discipline. In alignment with this, one of the International Council on Systems Engineering (INCOSE) strategic objectives is to accelerate this transformation. INCOSE is building a broad community that promotes and advances model-based methods to manage the complexity of systems today which seamlessly integrate computational algorithms and physical components across domains and traditional system boundaries. This paper addresses contextual drivers for transformation as well as obstacles, enablers, and INCOSE resources aligned with accelerating the transformation of Systems Engineering to a model-based discipline.

Introduction

Systems today are more interconnected, situationally aware, and intelligent than ever before. This shift is also changing the way we develop, manage, and interact with systems. The number of interactions is growing at an increasing rate, placing significant demands on organizations to improve system safety, security, and reliability. The National Science Foundation (NSF) notes that the challenges and opportunities of today's engineered systems which are built from, and depend upon, the seamless integration of computational algorithms and physical components, are both significant and far-reaching [1]. To address the challenges presented by these systems, the NSF has called for methods to conceptualize and design for the extensive interdependencies inherent in these systems within aerospace, automotive, energy, medical, manufacturing, and other sectors.

While system complexity advances and transforms the landscape, the Systems Engineering discipline is also experiencing a transformation—moving from a document-based to a model-based approach. This transformation, led by the International Council on Systems Engineering (INCOSE), is necessary to advance the discipline, manage complex emergent behaviors, and meet stakeholder and market needs.

Contextual Drivers

Figure 1 below outlines internal and external drivers influencing systems and their context. Collectively, these drivers influence both the discipline of Systems Engineering and the systems of interest that Systems Engineering produces. Internally, both systems complexity and digital transformation have a significant influence on the need and ability to transform the discipline of Systems Engineering. Likewise, externally the market is driving toward mass customization and technology adoption all while systems are rapidly connected to a vast array of other systems to improve stakeholder value. These interrelated systems and contextual influencers cut across all sectors, and they are driving profound change all around us.

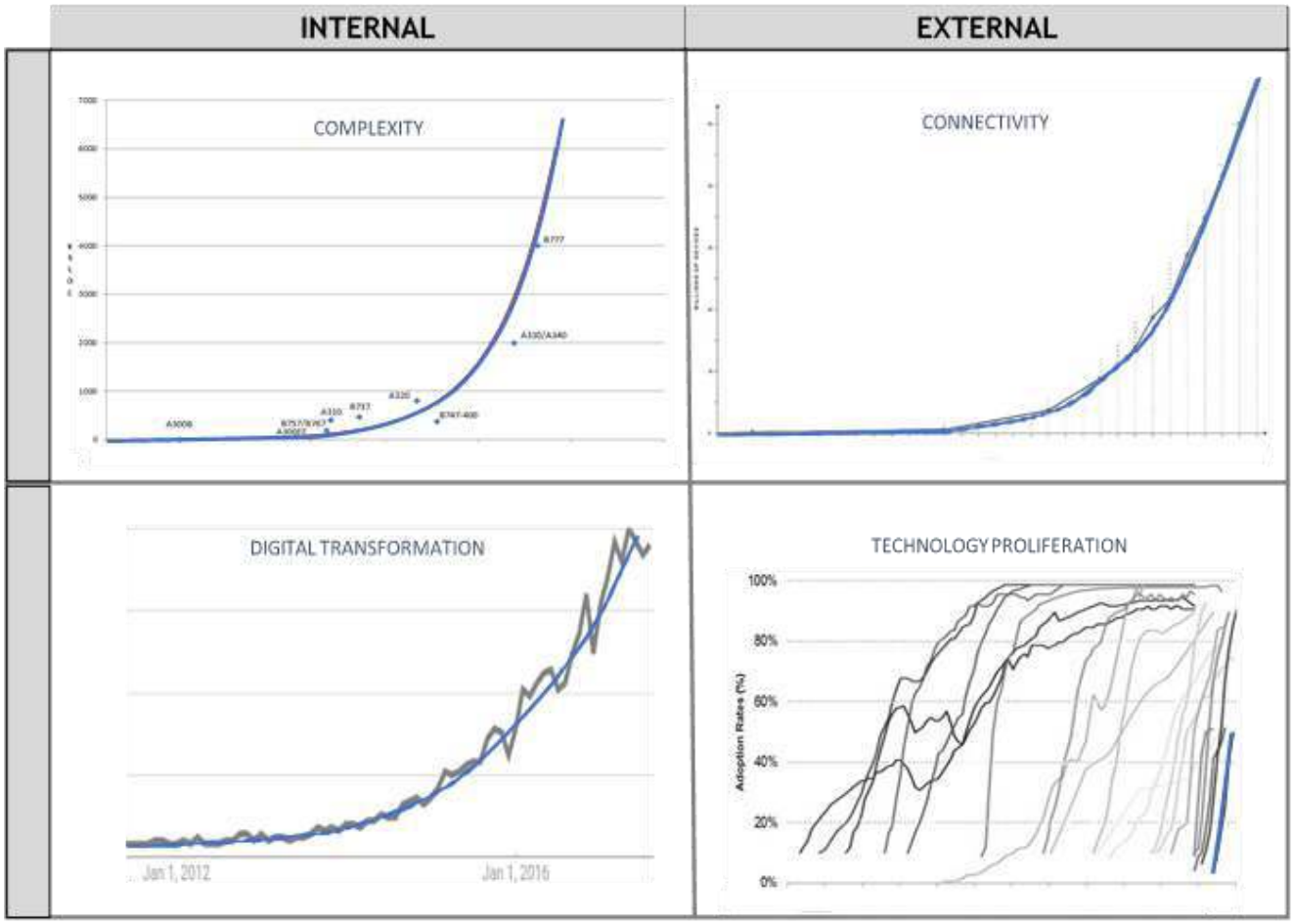

Figure 1: Growing internal and external system and contextual rate of change.

Consider the following quotes which are put forward as appropriate descriptions of the challenges we experience with our programs and systems.

- "Today more and more design problems are reaching insoluble levels of complexity."
- "At the same time that problems increase in quantity, complexity, and difficulty, they also change faster than before."

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

These quotes nicely summarize our context today, yet they were expressed over 50 years ago by Christopher Alexander in his book *Notes on the Synthesis of Form* [2]. These bullets are more applicable today than they were 50 years ago, and they will be even more applicable 50 years from now. Alexander offers a solution to these challenges in the form of "symbolic method" – or models which is the keystone to the transformation of Systems Engineering.

The Role of Digital Transformation

Digital Transformation is changing the fundamental nature of how organizations use and adopt digital technologies to create an order of magnitude change in value creation. This value can be expressed in more desirable products or in reduced resources to produce value. The lower left graphic of Figure 1 shows the rise in interest in Digital Transformation provided by Google Trends for the term "Digital Transformation". In like fashion there has been a similar growth in publications covering the topic of Digital Transformation. Many of these publications, however, emphasize the anticipated impact on business systems such as corporate websites and marketing strategies, sales efficiencies, front office processes, and the customer experience as outlined in *The Digital Imperative*, Boston Consulting Group 2015 [3]. In another publication, *The Digital Business Imperative* from Forrester, 401 global executives in companies with 250 or more employees identified e-commerce, marketing, and IT as the top three functions to be impacted most by digital technologies [4]. This same report from Forrester charted insurance and retail as the most likely to have digital technology drive strategy and disrupt their business. It also mapped engineering as the least likely out of a dozen different sectors to have digital technology drive strategy and disrupt their business. These and other publications noted that the most significant impact of digital technologies is when they are leveraged to impact the most substantive value creation aspects of an enterprise. For many organizations, the most significant value creation area and core capability is new product development or Systems Engineering – the aspect of the business most directly impacting innovation. Unfortunately, the cited publications and many other articles and publications do not address the significant impact and value of a broader systems approach, and its effect on the innovation process and role of modeling and engineering. **This paper proposes that transforming Systems Engineering through the application of systems modeling is the essential element to value creation via Digital Transformation. It is the area of greatest leverage and can drive revolutionary change in today's hyperconnected dynamic world.**

To be at the heart of this revolution requires that organizations understand contextual drivers and also leverage the major technological shifts, many of which have been identified by INCOSE, the World Economic Forum, and other sources. System modeling provides a virtual system synthesis - an integrated view of how evolving digital technologies impact designs and businesses. A model-based approach, however, requires a significant culture shift as well as a new mindset to operate within a data-centric digital

environment. The paper “Deep Shift: Technology Tipping Points and Societal Impact” by the Global Agenda Council, World Economic Forum [5] highlights Six Megatrends affecting the global economy. It describes the convergence of the Internet of Things (IoT), Artificial Intelligence (AI), 3D printing, and other megatrends which are truly reshaping the industry. The trends identified by the World Economic Forum and others are supported and enabled through model-based systems engineering methods, providing both new opportunities as well as ways to mitigate risks. Following are a few example trend areas and associated challenges and opportunities for how model-based systems engineering methods can help.

The *Internet of Things (IoT)* is a trend that has created a rapid explosion of interrelated systems and is a significant contributor to the increase in interconnectedness and system complexity. Impacts of the IoT are still only in the formative stages, and it continues to have exponential growth. This hyperconnectivity is ubiquitous, occurring across domains and with systems we use every day. **Systems Engineering is fundamentally about designing and managing interactions to drive state change and desired behaviors.** As IoT drives up the number of interactions, the application of model-based systems engineering becomes a necessary and logical method to fully express and manage these interactions and their associated risks and opportunities.

Another trend, *Data Science,* is also growing rapidly. The amount of data from instrumented and connected devices covers the full lifecycle from identification of stakeholder needs to system disposal. The amount of data organizations receive from systems is immense; in fact, it’s often overwhelming. It’s been said that many firms are drowning in data, and starving for information. The use of systems models to help put data in context, however, provides a great opportunity to improve engineering with robust feedback loops and ongoing learning and continual refinement of systems models and the opportunity to speed the overall engineering effort.

Cyber Security is another a rapidly growing need and trend aided by MBSE. As our system boundaries expand and more and more systems and subsystems become interconnected, new vulnerabilities and failure modes are often introduced. Here multidisciplinary models that bridge the cyber and physical domains are essential. Cyber security is often viewed from an enterprise IT perspective rather than an engineering viewpoint of a Cyber-Physical System. As more and more elaborate systems models are produced integrating both the cyber and physical aspects of a design, a more complete understanding of cyber security and safety is being provided. System models can easily be traversed to assess vulnerabilities, propagation paths, identify broker components, and more.

Artificial Intelligence (AI) may be the trend with the most hype given the strong movement toward autonomy. AI is being incorporated very rapidly into more and more domains and products. In 1997, IBM’s Deep Blue beat chess grand master, Gary Kasparov. In 2009, Garry Kasparov concluded that “weak human + machine + better process was superior to a strong computer alone and, even more, remarkable, is that it’s superior to a strong human + machine + inferior process.” He also said that “Today, for $50 you can buy a home PC program that will crush most grandmasters.” Design space, like chess, is a constrained space but rather than be limited by a board, pieces, and rules, designs are often constrained

by cost, schedule, technological advancements, and existing assets. Formal systems models provide a robust way to enable the use of AI, and when combined with human expert knowledge and robust systems engineering processes, they can provide a significant competitive advantage in the development and quality of complex systems.

The Role of System Complexity and Rate of Change

Systems complexity is growing at an unsettling rate with the number and density of interactions increasing dramatically. This increase is creating a web of interactions and related behaviors unlike what we have dealt with in the past. As an example, to demonstrate how interactions drive complexity, consider Figure 2 which presents two basic systems represented by nodes (N) and links (L). The first system on the left is the five-node system. To calculate how many potential links (PL) this system can have, we can use the equation provided in Figure 2, where N is the number of nodes representing system elements, i.e., subsystems or components, and the lines represent a simple connection or potential interaction without a type or direction. The number of unique configurations of this simple system is 2^{PL}. In comparing the 5-element system and 30 element system on the right, one can see how quickly the number of Unique Configurations (UC) grows with the increase in system elements and interactions. Surprisingly, an 18-element system with a network density of one, where every node is connected to every other node, has more unique configurations than the number of known atoms in the universe. While a network density of one is highly unlikely, this example does show how the rapid increase in instrumenting our systems and connecting them with external systems is impacting the complexity of systems management and our engineering efforts.

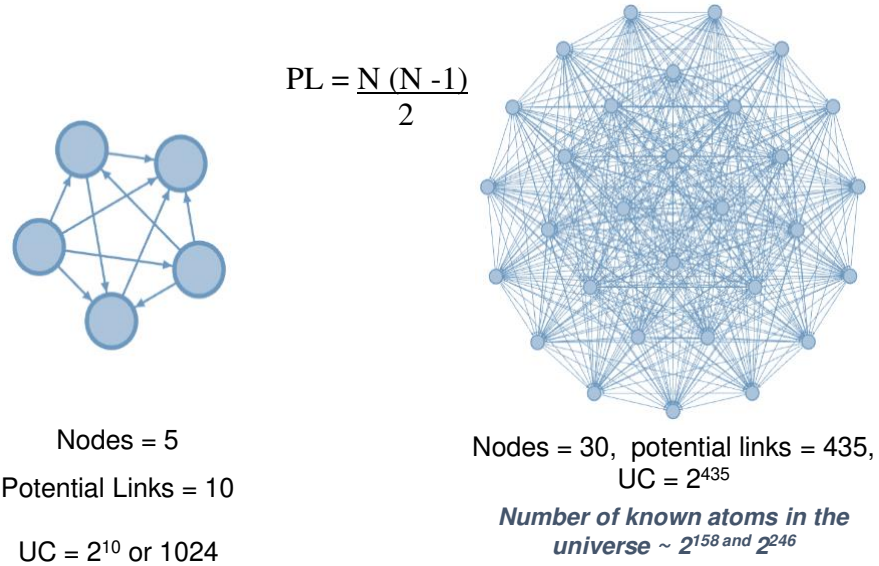

Figure 2: Example of how interaction density increases system complexity

Provided the significant shifts outlined above occur, especially the contextual changes associated with systems complexity and digital transformation, organizations will be seeking approaches which leverage

symbolic method, or formal models, to speed life-cycle management of systems and ensure a holistic view when making decisions. Accelerating this approach is the mission of INCOSE's transformation effort.

Systems Engineering Transformation

INCOSE: "The International Council on Systems Engineering (INCOSE) is a not-for-profit membership organization founded to develop and disseminate the interdisciplinary principles and practices that enable the realization of successful systems." [6] INCOSE's vision is "A better world through a systems approach." To meet this end, INCOSE produces state-of-the-art products that further the systems engineering discipline, leading to improvements in productivity, effectiveness, overall system performance, and ultimately increased stakeholder value. In today's new context MBSE is a must – just as outlined in INCOSE's vision 2025.

INCOSE Vision 2025: INCOSE's Vision 2025 addresses the movement to transform the practice of Systems Engineering to a model-based discipline. The vision states that "Systems Engineering will lead the effort to drive out unnecessary complexity through well-founded architecting and deeper system understanding; which can only be reasonably achieved through the use of models." [7] The vision goes on to describe a virtual engineering environment that will incorporate modeling, simulation, and visualization to support all aspects of systems engineering. The INCOSE Vision 2025 provides the Systems Engineering "From" and "To" states identified in the sidebar.

From:
Model-based systems engineering has grown in popularity as a way to deal with the limitations of document-based approaches but is still in an early stage of maturity similar to the early days of CAD/CAE.

To:
Formal systems modeling is standard practice… and is fully integrated with other engineering models. System models are adapted to the application domain and include a broad spectrum of models for representing all aspects of systems. The use of internet-driven knowledge representation and immersive technologies enables highly efficient and shared human understanding of systems in a virtual environment that spans the full life cycle from concept through development, manufacturing, operations, and support.

INCOSE Systems Engineering Strategic Objective: The systemic complexity of systems today demands a systems engineering approach. It requires a systems paradigm which is interdisciplinary, leverages principles common to all complex systems, and applies the requisite physics-based and mathematical models to represent them. It also requires a model-based approach to manage the immense amount of interrelated engineering data and information. "Model-Based Systems Engineering is the formalized application of modeling to support life cycle system engineering activities, beginning in the conceptual design phase and continuing throughout development and later life cycle phases." INCOSE also notes that "Modeling has always been an important part of systems engineering to support functional, performance, and other types of engineering analysis."

Systems Engineering Transformation, one of INCOSE's seven strategic objectives [8], is an essential objective for INCOSE and the systems engineering discipline. The INCOSE Systems Engineering Transformation strategic objective is provided below:

INCOSE accelerates the transformation of systems engineering to a model-based discipline.

In alignment with this objective, the application of model-based methods has increased dramatically in recent years, becoming a more standard practice for those who have established systems engineering capabilities. This shift in approach has been enabled by the continued maturity of standards for modeling languages, methodologies, and related technologies, including capability advancements made by tool vendors.

Figure 3 below outlines a vision, mission areas, goals, and objectives designed to help achieve INCOSE's strategic objective for transformation. The mission areas are targeted to:

1. Infuse INCOSE with model-based methods,

2. Engage Stakeholders external to INCOSE to better understand and share what is practiced and needed, and

3. Advance the Practice and share with stakeholders what is possible through the application and transformation to a model-based approach.

Vision	**Systems Engineering is acknowledged as a model based discipline**		
Mission	**INCOSE accelerates the transformation of systems engineering to a model-based discipline**		
Mission Area #	**1**	**2**	**3**
Mission Area	**Infuse INCOSE**	**Engage Stakeholders**	**Advance Practice**
Mission Area	**What can INCOSE Do?**	**What is practiced and needed?**	**What is possible?**
Goals	Infuse model based methods throughout INCOSE products, activities and WGs	Engage stakeholders to assess the current state of practice, determine needs and values of model based methods	Advance stakeholder community model based application and advance model based methods.
Objective 1 Foundations	Inclusion of model based content in INCOSE existing/new products (Vision, Handbook, SEBoK, Certification, Competency Model, etc.)	Define scope of model based systems engineering with MBE practice and broader modeling needs	Advance foundational art and science of modeling from and best practices across academia, industry/gov. and non profit.
Objective 2 Expand Reach	Expand reach within INCOSE of MBSE Workshop; highlight and infuse tech ops activities with more model based content (products, WGs etc.)	Identify, categorize and engage stakeholders and characterize their current practices, enablers and obstacles	Increase awareness of and about stakeholders outside SE discipline of what is possible with model based methods across domains and disciplines (tech/mgmt)
Objective 3 Collaborate	Outreach: Leverage MOUs to infuse model based content into PMI, INFORMS, NAFEMS, BIM, ASME and others, sponsoring PhD Students, standardization bodies, ABET	Build a community of Stakeholder Representatives to infuse model based advances into organizations practicing systems engineering.	Initiate, identify and integrate research to advance systems engineering as a model based discipline
Objective 4 Assessment/ Roadmap	Assess INCOSE's efforts (WG, Objectives, Initiatives etc.) for inclusion of model based methods across the Systems Modeling Assessment/Roadmap	Engage stakeholder community with Systems Modeling Assessment/ Roadmap to better understand the state of the practice of MBSE. Push and pull content from stakeholders (change agents and the "to be convinced")	Provide baseline assessment framework, Systems Modeling Roadmap, to create a concrete measure of current state of the art of what's possible/what's the potential.

Figure 3: INCOSE Systems Engineering Transformation Mission Areas, Goals, and Objectives

The Challenges of Transformation

The focus of INCOSE's strategic objective is to accelerate the transformation already underway. However, transformation, let alone accelerating it, has many challenges. The consulting firm McKinsey has noted that over 70 percent of complex, large-scale change programs fail to achieve their goals [9]. In their report on Transformation, McKinsey also found that this is primarily due to employee resistance and lack of management support. The Harvard Business Review (HBR) article notes similar challenges with Digital Transformation, citing organizational silos, legacy processes, and cultural resistance to change as key barriers. Figure 4 outlines the primary issues identified in the HBR article that tend to hold organizations back from achieving their transformation goals. Note also that these barriers apply to Digital Transformation leaders, followers, and laggards.

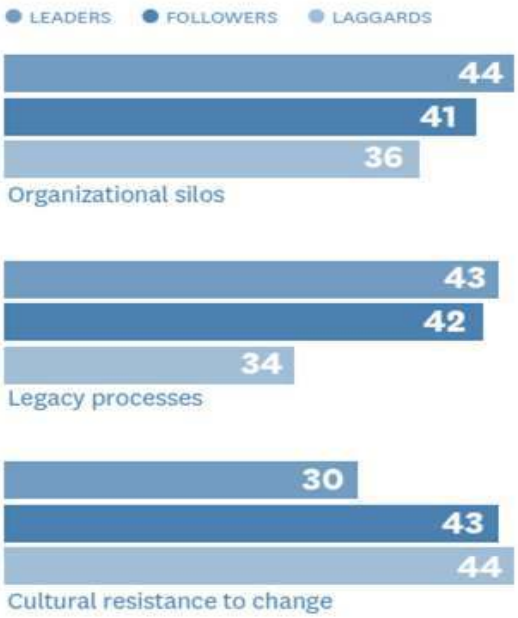

Figure 4: Barriers to Digital Transformation [viii]

Figure 5 [10] is an overlay of Gartner's Hype-Cycle[1] [11] and a graphic from Geoffrey Moore's Crossing the Chasm [11], with adoption from Roger's Innovation cycle [12]. These models frame the challenges of transformation well. They express the difficulty in managing expectations as shown in Gartner's Hype Cycle with its peak of inflated expectations and trough of disillusionment. Moore's Crossing the Chasm graphic identifies the roles in the transformation that stakeholders often take on, such as innovators, adopters, and laggards. The Transformation activities within INCOSE are aimed at supporting adoption and these transitions. This includes understanding and managing expectations as well as aiding those change agents who are navigating the chasm. INCOSE is working to produce products and communicate best practices and lessons learned to reduce inflated expectations, minimize the trough of disillusionment, and ultimately ramp up productivity in the application of systems engineering using formal models.

[1] Hype Cycle is a branded graphical presentation developed and used by the IT research and advisory firm, Gartner.

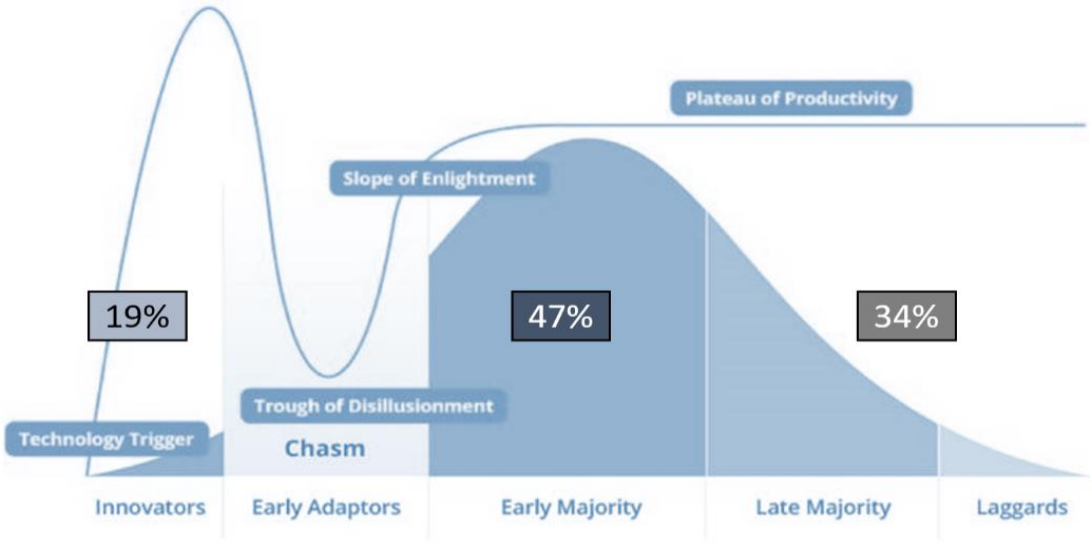

Figure 5: Gartner Hype-cycle and Moore/Rogers Crossing the Chasm

INCOSE and other sources have identified many enablers and barriers to transformation. Figure 6 outlines enablers, needs, and obstacles by categorical area identified by INCOSE's Corporate Advisory Board (CAB). Table 1 outlines a summary of enablers for Digital Transformation from the Harvard Business Review Analytic Services Report titled "Driving Digital Transformation: New Skills for Leaders, New Role for the CIO" [13].

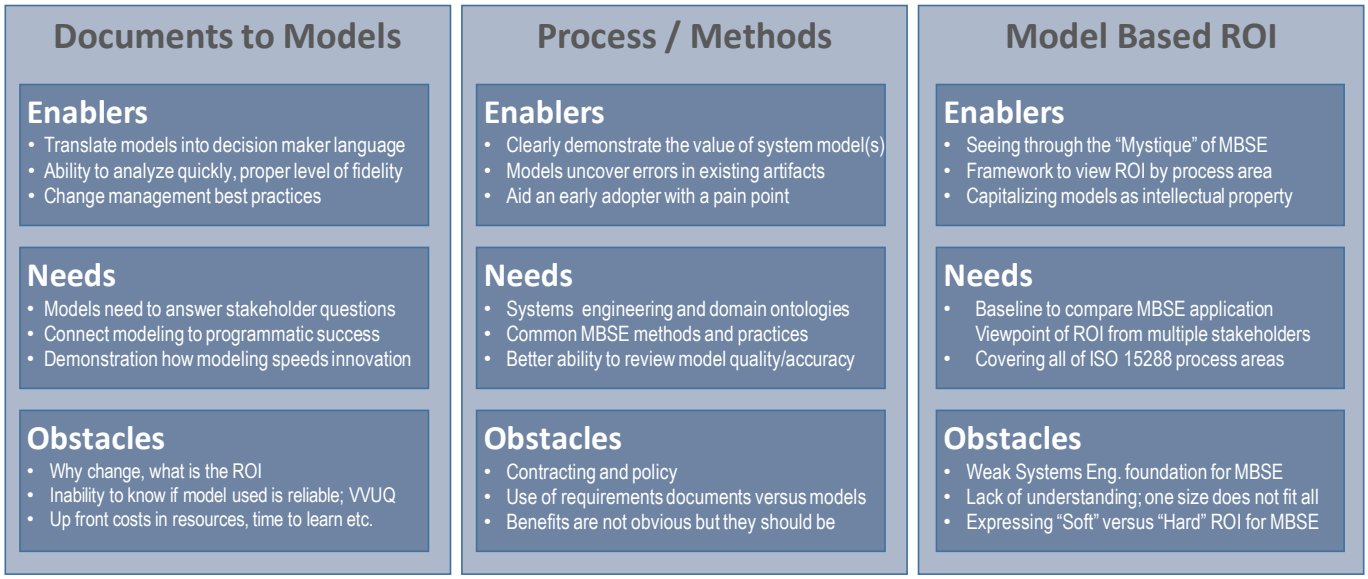

Figure 6: INCOSE Corporate Advisory Board Enablers, Needs and Obstacles for Model Based by categorical area

Table 1: Keys to the Digital Transformation

Keys to Digital Transformation
Digital leadership starts at the top
Engage in a discussion of trends
Think about agile
Use examples to make it real
Need a foundation of trust
Use KPIs for sharing knowledge
Break down walls wherever possible
Need digital coaches or masters
Create appropriate learning forums

INCOSE Activities Aligned to Transformation

INCOSE's Working Groups (WG) are one of the key organizational elements that produce technical products. INCOSE has nearly 50 working groups which are organized into four general areas: Application Domains, Analytic Enablers, Process Enablers, and Transformational Enablers. Several INCOSE Working Groups which have projects and products aligned with the Transformation Strategic Objective. Some of these WGs are noted in Table 2.

Table 2: MBSE Initiative Challenge Team Name

MBSE Initiative Challenge Team Name
Agile Systems & Systems Engineering
Lean Systems Engineering
Model-Based Concept Design
Object-Oriented SE Method
MBSE Patterns
System Science
Ontologies
Systems and Software Integration
Enterprise Systems
Tool Integration & Lifecycle Management
Digital Engineering Information Exchange
Very Small Entities (VSE)
Model Based Systems Engineering Initiative

The MBSE Initiative which resides within INCOSE's Transformational Enablers set of Working Groups is being used as an innovation incubator to kick-start innovative activities to advance the transformation of systems engineering and the application of model-based methods. Table 3 outlines three new Challenge Teams and one collaborative effort associated with the Patterns WG that were initiated by the efforts of WGs that are aligned with INCOSE's Systems Engineering Transformation strategic objective.

MBSE Initiative Challenge Team Name	Activities
Augmented Intelligence in Systems	How can machine learning and AI aid systems engineering and systems engineers in the innovation process
Digital Engineering Capabilities Assessment	Developing self-assessments and gap analysis, strategic planning, project progress aids aligned with the application of Digital Engineering
Production and Distribution Systems	Connecting models across the lifecycle to include production and logistics models and impacts of Industry 4.0, supply chain, logistics and more
VVUQ of models	Verification and Validation of Models – tied to ASME VV50 standards project is an essential element of trust in model and uncertainty quantification

Organizational Change Management

One of the most important areas in supporting the transformation of Systems Engineering to a model-based discipline is the need for organizational change management. As outlined under the subheading "The Challenges of Transformation" the ability to manage change within a project, program, enterprise or domain is the best predictor for a successful shift to a model-based paradigm and approach. There are many references to aid change agents, teams and organizations and Figure 7 below provides one perspective of the many dimensions of change that must be considered to obtain the order of magnitude improvements required to manage the complexity of systems today.

Figure 7: Factors to consider for transformation and organizational change management

John P. Kotter in his well-known book Leading Change [14] provides an Eight-Step Process for undertaking major change.

1. Creating a Sense of Urgency
2. Building a Guiding Coalition
3. Developing a Strategic Vision and Initiatives
4. Expanding the Network of Change Agents
5. Empowering Broad-Based Action
6. Generating Short-Term Wins
7. Consolidating Gains and Producing More Change
8. Instituting Change in the Culture

In Kotter's new book *Accelerate* he refines these principals and adds the concept of a "dual operating system". One operating system is characterized by management, hierarchy and driven toward efficiency. The other is characterized by leadership, networks and strategic acceleration and driven to innovate. These two operating systems align nicely with the System of Innovation framework used in INCOSE's Agile and Patterns Working Groups where we see the distinct roles of executing and managing systems development and managing knowledge and what is learned in execution. Both of these roles, or systems, are essential aspects to organizational change and transformation. Likewise, these also play important roles in the transformation of systems engineering to a model-based discipline.

Another important aspect of broad transformation is the inclusion of a broad stakeholder community that is multidisciplinary and represents the expanding spectrum of stakeholders included in our systems development activities. INCOSE is collaborating with many other organizations through its Corporate Advisory Board and numerous Memorandums of Understanding with other professional societies such as the Object Management Group (OMG), the International Association for the Engineering, Modeling, Analysis, and Simulation Community (NAFEMS), the International Society for the Systems Sciences (ISSS), the National Defense Industrial Association (NDIA), Military Operations Research Society (MORS), the Project Management Institute (PMI), and others.

As an example, INCOSE's collaboration with PMI has identified several dimensions of needed change to enable the integration of systems engineering and program management, particularly for complex projects. The book, *Integrating Program Management and Systems Engineering* (IPMSE) [15], is a collaboration of the International Council on Systems Engineering (INCOSE), the Project Management Institute (PMI), and the Consortium for Engineering Program Excellence (CEPE) at the Massachusetts (USA) Institute of Technology (MIT). The key message of the book is that the implementation of current systems engineering practice is not sustainable, which is to say that the state of systems engineering practice must improve or else face the consequences, such as a continued high rate of failures in developing complex systems.

Summary and Conclusions

INCOSE's Vision 2025 calls for Systems Engineering to lead the effort to drive out unnecessary complexity and to help stakeholders obtain a deeper system understanding. It calls for us to develop a virtual engineering environment with rich, dynamic visualization and analysis. It emphasizes the objective to obtain shared human understanding which is essential for true transformation. INCOSE is building a broad community that promotes and advances model-based methods to manage the complexity of systems today across all domains.

For this transformation to occur the systems engineering effort must provide value within the business context of increasing complexity and digital transformation. Change agents must lead and help teams and enterprises leverage enablers and overcome identified obstacles, especially those associated with organizational change management.

It is an exciting time for systems engineers and the discipline of Systems Engineering. We have entered into the age of systems, a time when systems thinking, systems engineering, and the ability to innovate and technically lead is essential. We are maturing model-based methods to help organizations address growing complexity, leveraging digital technologies, and capitalizing on new and emerging trends. Systems Engineering Transformation, and INCOSE's lead role in it, is the essential element in the broader digital transformation and in providing stakeholder value through complex systems development.

List of Acronyms Used in this Paper

Acronym	Explanation
3D	Three Dimensional
AI	Artificial Intelligence
ASME	American Society of Mechanical Engineers
CAD	Computer Aided Design
CAB	Corporate Advisory Board
CAM	Computer Aided Manufacturing
CEPE	Consortium for Engineering Program Excellence at MIT
CIO	Chief Information Officer
HBR	Harvard Business Review
INCOSE	International Council on Systems Engineering
IoT	Internet of Things
MBE	Model Based Engineering
MBSE	Model Based Systems Engineering
MIT	Massachusetts Institute of Technology
MOU	Memorandum of Understanding
NAFEMS	International Association for the Engineering Modeling, Analysis, and Simulation Community
NDIA	National Defense Industrial Association
NSF	National Science Foundation
OMG	Object Management Group
PMI	Project Management Institute
SE	Systems Engineering
UC	Unique Configurations
VV50	Verification and Validation 50
WG	Working Group

Acknowledgments

I would like to acknowledge the INCOSE Systems Engineering Transformation Team all of whom have contributed to the INCOSE Transformation effort. The core team includes Bill Schindel, Mark Sampson, Jeff Bergenthal, and Joachim Fuchs.

I would also like to acknowledge PPI's coverage of the key considerations in Integrating Systems Engineering and Program Management as an essential element of transforming the discipline of systems engineering. As we collectively seek to improve and transform systems engineering, we can ultimately change and affect the way our organizations conduct business to include how program management and systems engineering are more fully integrated and the positive outcomes associated with this integration. A short summary of each of the sixteen chapters of the book, *Integrating Program Management and Systems Engineering*, is provided in issues 54 – 70 of the *Project Performance International Systems Engineering Newsletter* (PPI SyEN). Use the following link to review the articles in these issues:

http://www.ppi-int.com/newsletter/systems-engineering-newsletter.php

References

[1] The National Science Foundation (NSF); Definition of Cyber-Physical Systems (CPS).

[2] Alexander, Christopher. *Notes on the Synthesis of Form*. Harvard University Press, 1964.

[3] The Digital Imperative, *BCG Perspectives,* Boston Consulting Group, 2015.

[4] *The Digital Business Imperative*, Forrester, February 15, 2017.

[5] Global Agenda Council, World Economic Forum. "Deep Shift: Technology Tipping Points and Societal Impact". *Survey Report*, September 2015.

[6] https://www.incose.org/about-incose

[7] INCOSE Vision 2025, 2014.

[8] https://www.incose.org/about-incose/strategic-objectives

[9] https://mckinsey.com/industries/retail/our-insights/the-how-of-transformation

[10] Hype Cycle, Chasm Combined Graphic: http://www.datameer.com/blog/big-data-analytics-perspectives/big-data-crossing-the-chasm-in-2013.html

[11] Moore, Geoffrey A. "Crossing the Chasm and Beyond" Strategic Management of Technology and Innovation, Third Edition, 1996.

[12] Rogers, Everett M. *Diffusion of Innovations*. New York, Free Press of Glencoe, 1962.

[13] Harvard Business Review Report: Driving Digital Transformation: New Skills for Leaders New Role for the CIO, 2015.

[14] Kotter, John P, "Leading Change" Harvard Business School Press, Boston MA, 1996.

[15] Rebentisch, Eric (Editor-in-Chief). *Integrating Program Management and Systems Engineering.* Hoboken, New Jersey USA: John Wiley & Sons, 2017.

About the Author

Troy Peterson, SSI Vice President and INCOSE Transformation lead, is a recognized leader in helping organizations to speed innovation and solve complex systems challenges. He has led the delivery of numerous complex systems and methodologies while at SSI, Booz Allen, and Ford Motor Company. His experience spans academic, non-profit, commercial, and government environments across all lifecycle phases. Troy received a BS in Mechanical Engineering from Michigan State University, an MS in Technology Management from Rensselaer Polytechnic Institute, and an advanced graduate certificate in Systems Design and Management from the Massachusetts Institute of Technology. He also holds INCOSE CSEP, PMI PMP, and ASQ Six Sigma Black Belt Certifications.

3. ARTICLES

3.1 Geriatric Issues of Aging Software

by

Capers Jones,

VP and CTO of Namcook Analytics LLC

Email: Capers.Jones3@gmail.com

Web: www.Namcook.com

Editor's note: PPI is once again delighted to provide subscribers important data-based conclusions based on Capers Jones' research that will help organizations improve engineering practice. The article is provided below, following a summary of it. We are thankful for Capers' generosity in sharing his research with the systems engineering community.

Summary of the observations, conclusions, and recommendations provided in this article:

Organizations which are proactive in using geriatric tools and services can spend less than 30% of their annual software budgets on various forms of maintenance, while organizations that have not used any of the geriatric tools and services can top 60% of their annual budgets on various forms of maintenance.

Every company and government organization should measure defect potentials and defect removal efficiency levels.

Roughly 7% of all defect repairs will contain a new defect that was not there before. For very complex and poorly structured applications, these bad-fix injections have topped 20%.

Studies by corporations found that error-prone modules were endemic in the software domain.

It is possible to surgically remove error-prone modules once they are identified. It is also possible to prevent them from occurring. A combination of defect measurements, static analysis of all legacy code, formal design inspections, formal code inspections, and formal testing and test-coverage analysis have proven to be effective in preventing error-prone modules from coming into existence.

If the author's clients are representative of the U.S. as a whole, more than 50% of U.S. companies still do not utilize the CMMI at all. Of those who do use the CMMI, less than 15% are at level 3 or higher. This suggests that error-prone modules may exist in more than half of all large corporations and in a majority of state government software applications as well.

Once deployed, most software applications continue to grow at annual rates of between 5% and 10% of their original functionality. Some applications, such as Microsoft Windows, have increased in size by several hundred percent over a 10-year period.

The combination of continuous growth of new features coupled with continuous defect repairs tends to drive up the complexity levels of aging software applications.

The rate at which entropy or complexity increases is directly proportional to the initial complexity of the application. For example, if an application is released with an average cyclomatic complexity level of less than 10, it will tend to stay well-structured for at least five years of normal maintenance and enhancement changes.

Both bad-fix injections and error-prone modules tend to correlate strongly (although not perfectly) with high levels of complexity. A majority of error-prone modules have cyclomatic complexity levels of 10 or higher. Bad-fix injection levels for modifying high-complexity applications are often higher than 20%.

In the late 1990's a special kind of geriatric issue occurred which involved making simultaneous changes to thousands of software applications. These are known as "mass update" geriatric issues, for example, the Y2K problem.

Although normal counting of function points is not feasible for small updates and micro function points are still experimental, it is possible to use the "backfiring" method or converting counts of logical source code statements in to equivalent function points.

Porting is a form software modification that is difficult to count via function points because no new features are added.

At the top of the list of maintenance "best practices" is the utilization of full-time, trained maintenance specialists rather than turning over maintenance tasks to untrained generalists.

The positive impact from utilizing maintenance specialists is one of the reasons why maintenance outsourcing has been growing so rapidly. The maintenance productivity rates of some of the better maintenance outsource companies is roughly twice that of their clients prior to the completion of the outsource agreement. Thus even if the outsource vendor costs are somewhat higher, there can still be useful economic gains.

More than 80% of latent bugs found by users in software applications are reported against less than 20% of the modules. Once these modules are identified then they can be inspected, analyzed, and restructured to reduce their error content down to safe levels.

Table 7 summarizes the major factors that degrade software maintenance performance. Not only are error-prone modules troublesome, but many other factors can degrade performance too. For example, very complex "spaghetti code" is quite difficult to maintain safely. It is also troublesome to have maintenance tasks assigned to generalists rather than to trained maintenance specialists.

A very common situation which often degrades performance is lack of suitable maintenance tools, such as defect tracking software, change management software, test library software, and so forth. In general it is very easy to botch up maintenance and make it such a labor-intensive activity that few resources are left over for development work.

Unpaid overtime is very common among software maintenance and development personnel. In some companies it amounts to about 15% of the total work time. Because it is unpaid it is usually unmeasured. That means side by side comparisons of productivity rates or costs between groups with unpaid overtime and groups without will favor the group with unpaid overtime because so much of their work is uncompensated and hence invisible. This is a benchmarking trap for the unwary. Because excessive overtime is psychologically harmful if continued over long periods, it is unfortunate that unpaid overtime tends to be ignored when benchmark studies are performed.

Given the enormous amount of effort that is now being applied to software maintenance, and which will be applied in the future, it is obvious that every corporation should attempt to adopt maintenance "best practices" and avoid maintenance "worst practices" as rapidly as possible.

Probably the best of the older "standard" methodologies for maintenance and enhancements work are IBM's Rational Unified Process (RUP) and the pair of methods developed by Watts Humphrey and now endorsed by the Software Engineering Institute: Team Software Process (TSP) and Personal Software Process (PSP).

Every software legacy application should have test coverage tools and also cyclomatic complexity tools available and used often.

Recommendation

Given the enormous efforts and costs devoted to software maintenance, every organization/company should evaluate and consider best practices for maintenance, and should avoid worst practices if at all possible.

Article

Abstract

Software has been a mainstay of business and government operations for more than 60 years. As a result, all large enterprises utilize aging software in significant amounts. Some Fortune 500 companies exceed 10,000,000 function points or 500,000,000 in the total volume of their corporate software portfolios. Much of this software is now more than 15 years old, and some important aging applications such as air traffic control are more than 35 years old.

Maintenance of aging software tends to become more difficult and expensive year by year since annual updates gradually destroy the original structure of the applications and increase its entropy[2] and cyclomatic complexity[3]. **On average the cyclomatic complexity of aging software applications increases by as much as 1% per calendar year due to numerous minor and a few major changes.**

Aging software also becomes larger each year as changes increase function point totals and also total volumes of source code. Aging software often has large inclusions of “dead code” - code that used to perform a useful function but which has been bypassed by newer changes. Dead code tends to make maintenance more complex and harder to perform.

Another attribute of aging software is that of a steady increase in “bad fix” injections. Bad fixes are new bugs accidentally included in repairs of reported bugs. The U.S. average rate for bad-fix injections is about 1.50% for new applications, but this rate increases steadily over time as system structure decays and becomes more complex. After 20 years of continuous usage, software bad fix injections can top 7.00%. In a few extreme cases bad fixes on aging legacy applications have topped 20.00% per year.

Aging software may also contain troublesome regions with very high error densities called “error-prone modules.” Error prone modules are so complex and troublesome that they need to be surgically removed and replaced by modern error-free code segments.

---

[2] Entropy is the tendency of systems to destabilize and become more chaotic over time. Entropy exists in all complex systems, including software.

[3] Cyclomatic complexity is a software metric (measurement), used to indicate the complexity of a program. It is a quantitative measure of the number of linearly independent paths through a program's source code. A discussion of cyclomatic complexity. A discussion of cyclomatic complexity is available in Cyclomatic Complexity Revisited: An Empirical Study of Software Development and Maintenance, Forgotten Books, February 27, 2018.

There are also sociological issues with maintenance of aging software. Many software engineers prefer to work on new development because it is more glamorous. Also, some companies unwisely pay maintenance engineers lower salaries than development engineers. Because maintenance is less popular than new development, it has become a major field of software outsourcing. Over 50% of aging U.S. software applications are now maintained by outsource vendors rather than by an organization's own software staff. In general, maintenance outsourcing has been successful for both clients and outsource vendors, some of whom specialize in "managed maintenance."

(Some leading companies such as IBM pay development and maintenance personnel equally, and allow transfers back and forth.)

Many popular methodologies such as agile are ineffective for handling updates to aging legacy applications. Technologies that can be useful for providing geriatric care to aging legacy software include frequent and widespread use of static analysis tools, manual inspections of critical code segments, and identification and surgical removal of error-prone modules. Maintenance-friendly methodologies such as team software process (TSP) are also useful.

Introduction

In today's world of 2018, more than 50% of the global software population is engaged in modifying existing applications rather than writing new applications. This fact by itself should not be a surprise, because whenever an industry has more than 50 years of product experience the personnel who repair existing products tend to outnumber the personnel who build new products. For example there have long been more automobile mechanics in the United States who repair automobiles than there are personnel employed in building new automobiles.

The imbalance between software development and maintenance is opening up new business opportunities for software outsourcing groups. It is also generating a significant burst of research into tools and methods for improving software maintenance performance.

Maintenance and renovation of legacy software plays a major role in "brownfield" development, which involves carefully building new applications in an environment where they will need to interface with scores of older applications.

Brownfield development is analogous to putting up an office building in an urban environment where the site is surrounded by other office buildings whose occupants cannot be disrupted, nor the surrounding structures damaged during new construction.

What is Software Maintenance?

The word "maintenance" is surprisingly ambiguous in a software context. In normal usage it can span some 23 forms of modification to existing applications. The two most common meanings of the word maintenance include: 1) Defect repairs; and 2) Enhancements or adding new features to existing software applications.

Although software enhancements and software maintenance in the sense of defect repairs are usually funded in different ways and have quite different sets of activity patterns associated with them, many companies lump these disparate software activities together for budgets and cost estimates.

The author does not recommend the practice of aggregating defect repairs and enhancements, but this practice is very common. Consider some of the basic differences between enhancements or adding new features to applications and maintenance or defect repairs as shown in Table 1:

Table 1: Key Differences between Maintenance and Enhancements

	Enhancements (New features)	**Maintenance (Defect repairs)**
Funding source	Clients	Absorbed
Requirements	Formal	None
Specifications	Formal	None
Inspections	Formal	None
User documentation	Formal	None
New function testing	Formal	None
Regression testing	Formal	Minimal

Because the general topic of "maintenance" is so complicated and includes so many different kinds of work, some companies merely lump all forms of maintenance together and use gross metrics such as the overall percentage of annual software budgets devoted to all forms of maintenance summed together.

This method is crude, but can convey useful information. **Organizations which are proactive in using geriatric tools and services can spend less than 30% of their annual software budgets on various forms of maintenance, while organizations that have not used any of the geriatric tools and services can top 60% of their annual budgets on various forms of maintenance.**

The kinds of maintenance tools used by lagging, average, and leading organizations are shown in Table 2. Table 2 is part of a larger study that examined many different kinds of software engineering and project management tools.

It is interesting that the leading companies in terms of maintenance sophistication not only use more tools than the laggards, but they use more of their features as well. The function point values in Table 2 refer to the capabilities of the tools that are used in day to day maintenance operations. The leaders not only use more tools, but they do more with them.

Table 2: Numbers and Size Ranges of Maintenance Engineering Tools (Size data expressed in terms of function point metrics)

Maintenance Engineering	**Lagging**	**Average**	**Leading**
Reverse engineering		1,000	3,000
Reengineering		1,250	3,000
Code static analysis			1,500

Configuration control	500	1,000	2,000
Test support		500	1,500
Customer support		750	1,250
Debugging tools	750	750	1,250
Defect tracking	500	750	1,000
Complexity analysis			1,000
Mass update search engines		500	1,000
Function point subtotal	*1,750*	*6,500*	*16,500*
Number of tools	*3*	*8*	*10*

Before proceeding, let us consider 24 discrete topics that are often coupled together under the generic term “maintenance” in industry discussions, but which are actually quite different in many important respects:

Table 3 Major Kinds of Work Performed Under the Generic Term “Maintenance”

1. Major Enhancements (new features of > 50 function points)
2. Average enhancements (new features of 5 to 50 function points)
3. Minor Enhancements (new features of < 5 function points)
4. Maintenance (repairing customer defects for good will and pro bono)
5. Warranty repairs (repairing defects under formal contract of for a fee)
6. Customer support (responding to client phone calls or problem reports)
7. Error-prone module removal (eliminating very troublesome code segments)
8. Mandatory changes (required or statutory changes such as new tax laws)
9. Complexity or structural analysis (charting control flow plus complexity metrics)
10. Code restructuring (reducing cyclomatic and essential complexity)
11. Optimization (increasing performance or throughput)
12. Migration (moving software from one platform to another)
13. Conversion (Changing the interface or file structure)
14. Reverse engineering (extracting latent design information from code)
15. Reengineering (transforming legacy application to modern forms)
16. Dead code removal (removing segments no longer utilized)
17. Dormant application elimination (archiving unused software)
18. Internationalization (modifying software for international use)
19. Mass updates such as Euro or Year 2000 Repairs
20. Refactoring, or reprogramming applications to improve clarity
21. Retirement (withdrawing an application from active service)
22. Field service (sending maintenance members to client locations)

23. Reporting bugs or defects to software vendors

24. Installing updates received from software vendors

Although the 24 maintenance topics are different in many respects, they all have one common feature that makes a group discussion possible: They all involve modifying an existing application rather than starting from scratch with a new application.

Each of the 24 forms of modifying existing applications has a different reason for being carried out. However, it often happens that several of them take place concurrently. For example, enhancements and defect repairs are very common in the same release of an evolving application. There are also common sequences or patterns to these modification activities. For example, reverse engineering often precedes reengineering, and the two occur so often together as to essentially comprise a linked set. For releases of large applications and major systems, the author has observed from six to ten forms of maintenance all leading up to the same release!

Geriatric Problems of Aging Software

Once software is put into production, it continues to change in three important ways:

1. Latent defects still present at release must be found and fixed after deployment.
2. Applications continue to grow and add new features at a rate of between 5% and 10% per calendar year, due either to changes in business needs or to new laws and regulations, or both.
3. The combination of defect repairs and enhancements tends to gradually degrade the structure and increase the complexity of the application. The term for this increase in complexity over time is called “entropy.” The average rate at which software entropy increases is about 1% to 3% per calendar year.

Because software defect removal and quality control are imperfect, there will always be bugs or defects to repair in delivered software applications. The current U.S. average for defect removal efficiency is only about 90% of the bugs or defects introduced during development and defect removal efficiency has only improved slowly over more than 20 years. The actual values are about 4.5 bugs per function point created during development. But the range is from < 2.5 to > 6.5 bugs per function point. If 90% of these were found before release, about 0.25 to 0.65 bugs per function point would be released to customers.

However, best-in-class organizations combine low defect potentials with Defect Removal Efficiency (DRE) levels approaching and sometimes exceeding 99%. Clearly maintenance costs will be much lower for software with 99% DRE than for software that is below 90% DRE. **Every company and government organization should measure defect potentials and defect removal efficiency levels.**

Since defect potentials tend to rise with the overall size of the application, and since defect removal efficiency levels tend to decline with the overall size of the application, the overall volume of latent defects delivered with the application rises with size. **This explains why super-large applications in the range**

of 100,000 function points, such as Microsoft Windows and many Enterprise Resource Planning (ERP) applications may require years to reach a point of relative stability. These large systems are delivered with thousands of latent bugs or defects.

Not only is software deployed with a significant volume of latent defects, but a phenomenon called "bad fix injection" has been observed for more than 50 years. **Roughly 7% of all defect repairs will contain a new defect that was not there before. For very complex and poorly structured applications, these bad-fix injections have topped 20%.**

Even more alarming, once a bad fix occurs it is very difficult to correct the situation. Although the U.S. average for initial bad-fix injection rates is about 7%, the secondary injection rate against previous bad fixes is about 15% for the initial repair and 30% for the second. A string of up to five consecutive bad fixes has been observed, with each attempted repair adding new problems and failing to correct the initial problem. Finally the 6th repair attempt was successful.

In the 1970's the IBM Corporation did a distribution analysis of customer-reported defects against their main commercial software applications. **The IBM personnel involved in the study, including the author, were surprised to find that defects were not randomly distributed through all of the modules of large applications.**

In the case of IBM's main operating system, about 5% of the modules contained just over 50% of all reported defects. The most extreme example was a large data base application, where 31 modules out of 425 contained more than 60% of all customer-reported bugs. These troublesome areas were known as "Error-Prone Modules" (EPM).

Similar studies by other corporations such as AT&T and ITT found that error-prone modules were endemic in the software domain. More than 90% of applications larger than 5,000 function points were found to contain error-prone modules in the 1980's and early 1990's. Summaries of the error-prone module data from a number of companies was published in the author's book Software Quality: Analysis and Guidelines for Success.

Fortunately, it is possible to surgically remove error-prone modules once they are identified. It is also possible to prevent them from occurring. A combination of defect measurements, static analysis of all legacy code, formal design inspections, formal code inspections, and formal testing and test-coverage analysis have proven to be effective in preventing error-prone modules from coming into existence.

NOTE: The Software Risk Master (SRM) tool of Namcook Analytics LLC can predict defect potentials, Defect Removal Efficiency (DRE) and also Error-Prone Modules (EPM) and bad-fix injections. It can also predict multiple years of maintenance, enhancements, and customer support. The SRM default period is three years after release, since normal business changes make longer predictions uncertain.

Today in 2018, error-prone modules are almost nonexistent in organizations that are higher than level 3 on the capability maturity model integration (CMMI) of the Software Engineering Institute. However they

remain common and troublesome for level 1 organizations, and for organizations that lack sophisticated quality measurements and quality control.

If the author's clients are representative of the U.S. as a whole, more than 50% of U.S. companies still do not utilize the CMMI at all. Of those who do use the CMMI, less than 15% are at level 3 or higher. This suggests that error-prone modules may exist in more than half of all large corporations and in a majority of state government software applications as well.

Once deployed, most software applications continue to grow at annual rates of between 5% and 10% of their original functionality. Some applications, such as Microsoft Windows, have increased in size by several hundred percent over a 10-year period.

The combination of continuous growth of new features coupled with continuous defect repairs tends to drive up the complexity levels of aging software applications. Structural complexity can be measured via metrics such as cyclomatic and essential complexity using a number of commercial tools. If complexity is measured on an annual basis and there is no deliberate attempt to keep complexity low, the rate of increase is between 1% and 3% per calendar year.

However, and this is an important fact, the rate at which entropy or complexity increases is directly proportional to the initial complexity of the application. For example if an application is released with an average cyclomatic complexity level of less than 10, it will tend to stay well-structured for at least five years of normal maintenance and enhancement changes.

But if an application is released with an average cyclomatic complexity level of more than 20, its structure will degrade rapidly and its complexity levels might increase by more than 2% per year. The rate of entropy and complexity will even accelerate after a few years.

As it happens, both bad-fix injections and error-prone modules tend to correlate strongly (although not perfectly) with high levels of complexity. A majority of error-prone modules have cyclomatic complexity levels of 10 or higher. Bad-fix injection levels for modifying high-complexity applications are often higher than 20%.

In the late 1990's a special kind of geriatric issue occurred which involved making simultaneous changes to thousands of software applications. The first of these "mass update" geriatric issues was the deployment of the Euro, which required changes to currency conversion routines in thousands of applications. The Euro was followed almost immediately by the dreaded year 2000 or Y2K problem (6), which also involved mass updates of thousands of applications. More recently in March of 2007 another such issue occurred when the starting date of daylight savings time was changed.

Future mass updates will occur later in the century, when it may be necessary to add another digit to telephone numbers or area code. Yet another and very serious mass update will occur if it becomes necessary to add digits to social security numbers in the second half of the 21st century. There is also the potential problem of the UNIX time clock expiration in 2038.

Metrics Problems with Small Maintenance Projects

There are several difficulties in exploring software maintenance costs with accuracy. One of these difficulties is the fact that maintenance tasks are often assigned to development personnel who interleave both development and maintenance as the need arises. This practice makes it difficult to distinguish maintenance costs from development costs because management does not require recording how time is spent.

Another and very significant problem is the fact that a great deal of software maintenance consists of making very small changes to software applications. Quite a few bug repairs may involve fixing only a single line of code. Adding minor new features such as perhaps a new line-item on a screen may require less than 50 source code statements.

These small changes are below the effective lower limit for counting function point metrics. The function point metric includes weighting factors for complexity, and even if the complexity adjustments are set to the lowest possible point on the scale, it is still difficult to count function points below a level of perhaps 15 function points.

An experimental method called “micro function points” is in development for small maintenance changes and bug repairs. This method is similar to standard function points, but drops down to three decimal places of precision. Thus changes that involve only a fraction of a standard IFPUG function point can be measured. The micro function point method should become available by 2016.

Of course the work of making a small change measured with micro function points may be only an hour or less. But if as many as 10,000 such changes are made in a year, the cumulative costs are not trivial. Micro function points are intended to eliminate the problem that small maintenance updates have not been subject to formal economic analysis.

Quite a few maintenance tasks involve changes that are either a fraction of a function point, or may at most be less than 10 function points or about 1000 COBOL source code statements. **Although normal counting of function points is not feasible for small updates and micro function points are still experimental, it is possible to use the “backfiring” method or converting counts of logical source code statements in to equivalent function points.** For example, suppose an update requires adding 100 COBOL statements to an existing application. Since it usually takes about 105 COBOL statements in the procedure and data divisions to encode 1 function point, it can be stated that this small maintenance project is “about 1 function point in size.”

If the project takes one work day consisting of six hours, then at least the results can be expressed using common metrics. In this case, the results would be roughly “6 staff hours per function point.” If the reciprocal metric “function points per staff month” is used, and there are 20 working days in the month, then the results would be “20 function points per staff month.”

Metrics Problems with ERP Maintenance

Many large corporations use Enterprise Resource Planning (ERP) packages such as SAP, Oracle, PeopleSoft, J.D. Edwards, and many others. These packages are large and complex systems in the size range of 200,000 function points and higher.

Worse, dozens or even hundreds of legacy applications need to be ported into ERP packages when they are deployed. **Porting is a form software modification that is difficult to count via function points because no new features are added.**

Best and Worst Practices in Software Maintenance

Because maintenance of aging legacy software is very labor intensive it is quite important to explore the best and most cost-effective methods available for dealing with the millions of applications that currently exist. The sets of best and worst practices are not symmetrical. For example, the practice that has the most positive impact on maintenance productivity is the use of trained maintenance experts. However, the factor that has the greatest negative impact is the presence of "error-prone modules" in the application that is being maintained.

Table 4 illustrates a number of factors which have been found to exert a beneficial positive impact on the work of updating aging applications and shows the percentage of improvement compared to average results.

At the top of the list of maintenance "best practices" is the utilization of full-time, trained maintenance specialists rather than turning over maintenance tasks to untrained generalists. Trained maintenance specialists are found most often in two kinds of companies: 1) Large systems software producers such as IBM; and 2) Large maintenance outsource vendors. The curricula for training maintenance personnel can includes more than a dozen topics and the training periods range from two weeks to a maximum of about four weeks.

Table 4: Impact of Key Adjustment Factors on Maintenance (Sorted in order of maximum positive impact)

Maintenance Factors	Plus Range
Maintenance specialists	35%
High staff experience	34%
Table-driven variables and data	33%
Low complexity of base code	32%
Test coverage tools and analysis	30%
Static analysis of all legacy code	29%
Reengineering tools	27%
High level programming languages	25%
Reverse engineering tools	23%
Complexity analysis tools	20%
Defect tracking tools	20%

"Mass update" specialists	20%
Automated change control tools	18%
Unpaid overtime	18%
Quality measurements	16%
Formal base code inspections	15%
Regression test libraries	15%
Excellent response time	12%
Annual training of > 10 days	12%
High management experience	12%
HELP desk automation	12%
No error prone modules	10%
On-line defect reporting	10%
Productivity measurements	8%
Excellent ease of use	7%
User satisfaction measurements	5%
High team morale	5%
Sum	503%

Since training of maintenance specialists is the top factor, Table 5 shows a modern maintenance curriculum such as those found in large maintenance outsource companies.

Table 5: Sample Maintenance Curricula for Companies Using Maintenance Specialists

Software Maintenance Courses	Days	Sequence
Error-Prone Module Removal	2.00	1
Complexity Analysis and Reduction	1.00	2
Reducing Bad Fix Injections and error-prone modules (EPM)	1.00	3
Defect Reporting and Analysis	0.50	4
Change Control	1.00	5
Configuration Control	1.00	6
Software Maintenance Workflows	1.00	7
Mass Updates to Multiple Applications	1.00	8
Maintenance of COTS Packages	1.00	9
Maintenance of ERP Applications	1.00	10
Regression Testing, static analysis, and legacy defect removal	2.00	11
Test Library Control	2.00	12
Test Case Conflicts and Errors	2.00	13
Dead Code Isolation	1.00	14
Function Points for Maintenance	0.50	15
Reverse Engineering	1.00	16
Reengineering	1.00	17
Refactoring	0.50	18
Maintenance of Reusable Code	1.00	19
Object-Oriented Maintenance	1.00	20
Maintenance of Agile and Extreme Code	1.00	21
TOTAL	23.50	

The positive impact from utilizing maintenance specialists is one of the reasons why maintenance outsourcing has been growing so rapidly. The maintenance productivity rates of some of the better maintenance outsource companies is roughly twice that of their clients prior to the completion of the outsource agreement. Thus, even if the outsource vendor costs are somewhat higher, there can still be useful economic gains.

Let us now consider some of the factors which exert a negative impact on the work of updating or modifying existing software applications. Note that the top-ranked factor which reduces maintenance productivity, the presence of error-prone modules, is very asymmetrical. The absence of error-prone modules does not speed up maintenance work, but their presence definitely slows down maintenance work.

In general more than 80% of latent bugs found by users in software applications are reported against less than 20% of the modules. Once these modules are identified then they can be inspected, analyzed, and restructured to reduce their error content down to safe levels.

Table 6 summarizes the major factors that degrade software maintenance performance. Not only are error-prone modules troublesome, but many other factors can degrade performance too. For example, very complex "spaghetti code" is quite difficult to maintain safely. It is also troublesome to have maintenance tasks assigned to generalists rather than to trained maintenance specialists.

A very common situation which often degrades performance is lack of suitable maintenance tools, such as defect tracking software, change management software, test library software, and so forth. In general it is very easy to botch up maintenance and make it such a labor-intensive activity that few resources are left over for development work.

Table 6: Impact of Key Adjustment Factors on Maintenance

(Sorted in order of maximum negative impact)

Maintenance Factors	Minus Range
Error prone modules	-50%
Embedded variables and data	-45%
Staff inexperience	-40%
High complexity of base code	-30%
Lack of test coverage analysis	-28%
Manual change control methods	-27%
Low level programming languages	-25%
No defect tracking tools	-24%
No "mass update" specialists	-22%
No static analysis of legacy code	-18%
No quality measurements	-18%
No maintenance specialists	-18%
Poor response time	-16%

Management inexperience	-15%
No base code inspections	-15%
No regression test libraries	-15%
No HELP desk automation	-15%
No on-line defect reporting	-12%
No annual training	-10%
No code restructuring tools	-10%
No reengineering tools	-10%
No reverse engineering tools	-10%
No complexity analysis tools	-10%
No productivity measurements	-7%
Poor team morale	-6%
No user satisfaction measurements	-4%
No unpaid overtime	0%
Sum	-500%

The last factor, or lack of unpaid overtime, deserves a comment. Unpaid overtime is very common among software maintenance and development personnel. In some companies it amounts to about 15% of the total work time. Because it is unpaid it is usually unmeasured. That means side by side comparisons of productivity rates or costs between groups with unpaid overtime and groups without will favor the group with unpaid overtime because so much of their work is uncompensated and hence invisible. This is a benchmarking trap for the unwary. Because excessive overtime is psychologically harmful if continued over long periods, it is unfortunate that unpaid overtime tends to be ignored when benchmark studies are performed.

Given the enormous amount of effort that is now being applied to software maintenance, and which will be applied in the future, it is obvious that every corporation should attempt to adopt maintenance "best practices" and avoid maintenance "worst practices" as rapidly as possible.

Methodologies that are Maintenance-Strong and Maintenance-Weak

There are about 60 named software development methodologies in 2018 as shown in the author's new book by CRC Press, Software Methodologies: A Quantitative Guide. The majority of these were developed mainly for new development or "Greenfield" applications and are "maintenance weak." However some methodologies were envisioned as providing lifetime support for software applications and therefore are "maintenance strong" or effective in "brownfield" projects.

Unfortunately the world's most popular methodology, Agile, is in the "maintenance-weak" category. It was designed almost exclusively for new development and has no solid features for dealing with legacy applications. A majority of agile users revert to waterfall for their maintenance work.

Probably the best of the older "standard" methodologies for maintenance and enhancements work are IBM's Rational Unified Process (RUP) and the pair of methods developed by Watts Humphrey

and now endorsed by the Software Engineering Institute: Team Software Process (TSP) and Personal Software Process (PSP).

However, because maintenance is not as exciting as new development, it has become a major growth industry for outsource vendors. Quite a few of these have developed custom or proprietary maintenance methodologies of their own, such as the "managed maintenance" approach developed by Computer Aid Inc. (CAI). Other major outsource vendors with large maintenance contracts include IBM, Accenture, and CSC but there are many others.

Some of the tool suites used by these maintenance outsource companies include the features of:

1. Maintenance work benches
2. Code restructuring tools
3. Translators from old languages (i.e. COBOL) to modern languages (i.e. Java)
4. Data mining tools for extracting lost requirements and algorithms from old code
5. Automatic function point counting of legacy applications
6. Cyclomatic complexity analyzers
7. Static analysis tools for locating hidden bugs in legacy code
8. Test coverage tools
9. Security analysis tools to guard against cyber attacks
10. Redocumentation tools to refresh obsolete requirements
11. Estimating tools such as SRM that predict costs and feature growth

There are also a number of off-shore outsource vendors who promise lower costs than U.S. vendors. While this may be true, off-shore work is quite a bit more complicated due to large time-zone differences and also to the fact that almost daily contact may be needed between the maintenance teams and the legacy software owners.

Among the author's clients in the Fortune 500 category, about 65% of maintenance work is farmed out to vendors and only 35% stays in house. Maintenance work performed in-house includes mainly mission-critical applications or those that process classified or highly confidential data such as financial records or competitive information.

Customer Support: A Major Maintenance Weakness

Many readers of this report have probably tried to get customer support from software vendors. Some companies don't even provide contacts with customer support personnel; others depend upon user groups or forums. Very few companies are ranked "good" or "excellent" for customer support, with Apple often heading the list and IBM usually cited.

The author's own experiences with contacting software vendors' customer support are not good: the average wait for telephone contact with a live person was more than 12 minutes. About half of the support personnel did not speak English well enough for me to understand their comments; about 25% did not understand the problem and had no effective solutions.

Internal software is usually better than commercial software, but even so, support is often an afterthought and assigned to development personnel rather than to dedicated support personnel.

There are five factors that influence how many customer support personnel will be needed for any given application:

1. The size of the application in function points.
2. The number of customers or users ranging from 1 person to millions of users.
3. The number of latent bugs or defects in the application when released.
4. The legal liabilities of the software vendor in case of bugs or failures.
5. The prestige and public image sought by the vendor vs. competitors.

Obviously big and complicated applications with hundreds or even thousands of bugs will need more support personnel than small, simple software packages with few bugs.

Note: Namcool Analytics' Software Risk Master (SRM) tool predicts numbers of customer support personnel for a three-year period after release. It also predicts numbers of bugs, numbers of incidents, numbers of customer calls for help, and other topics that are of significance to those responsible for geriatric care of released software.

Software Entropy and Total Cost of Ownership

The word "entropy" means the tendency of systems to destabilize and become more chaotic over time. Entropy is a term from physics and is not a software-related word. However entropy exists in all complex systems, including software. All known compound objects decay and become more complex with the passage of time unless effort is exerted to keep them repaired and updated. Software is no exception. The accumulation of small updates over time tends to gradually degrade the initial structure of applications and makes changes grow more difficult over time.

Table 7 illustrates the impact of entropy over a 10-year period. Each year accumulated changes will degrade the application's structure and raise cyclomatic complexity. Also, each year, the higher complexity will progressively lower testing Defect Removal Efficiency (DRE).

Entropy can be stopped or reversed using methods such as refactoring or restructuring. Also, frequent use of static analysis can help. **Every software legacy application should have test coverage tools and also cyclomatic complexity tools available and used often.**

For software applications entropy has long been a fact of life. If applications are developed with marginal initial quality control they will probably be poorly structured and contain error-prone modules. This means that every year, the accumulation of defect repairs and maintenance updates will degrade the original

structure and make each change slightly more difficult. Over time, the application will destabilize and "bad fixes" will increase in number and severity. Unless the application is restructured or fully refurbished, eventually it will become so complex that maintenance can only be performed by a few experts who are more or less locked into the application.

Table 7: Complexity and Entropy over time

Year	Cyclomatic Complexity	Testing DRE
2015	15	92%
2016	16	92%
2017	17	91%
2018	19	90%
2019	20	90%
2020	22	88%
2021	24	87%
2022	26	86%
2023	28	85%
2024	30	84%

By contrast, leading applications that are well structured initially can delay the onset of entropy. Indeed, well-structured applications can achieve declining maintenance costs over time. This is because updates do not degrade the original structure, as happens in the case of "spaghetti bowl" applications where the structure is almost unintelligible when maintenance begins.

The total cost of ownership of a software application is the sum of six major expense elements: 1) the initial cost of building an application; 2) the cost of enhancing the application with new features over its lifetime; 3) the cost of repairing defects and bugs over the application's lifetime; 4) The cost of customer support for fielding and responding to queries and customer-reported defects; 5) The cost of periodic restructuring or "refactoring" of aging applications to reduce entropy and thereby reduce bad-fix injection rates; 6) Removal of error-prone modules via surgical removal and redevelopment. This last expense element will only occur for legacy applications that contain error-prone modules.

Similar phenomena can be observed outside of software. If you buy an automobile that has a high frequency of repair as shown in Consumer Reports and you skimp on lubrication and routine maintenance, you will fairly soon face some major repair problems – usually well before 50,000 miles.

By contrast, if you buy an automobile with a low frequency of repair as shown in Consumer Reports and you are scrupulous in maintenance, you should be able to drive the car more than 100,000 miles without major repair problems. Indeed, some automobiles have been driven over 250,000 miles.

Summary

In every industry, maintenance tends to require more personnel than those building new products. For the software industry, the number of personnel required to perform maintenance is unusually large and may soon top 70% of all technical software workers. The main reasons for the high maintenance efforts in the software industry are the intrinsic difficulties of working with aging software (listed in Table 7). Special factors such as "mass updates" that began with the roll-out of the Euro and the year 2000 problem are also geriatric issues.

References

A. Books by Capers Jones that discuss software maintenance.

Jones, Capers, Software Methodologies: A Quantitative Guide; CRC Press, 2017.

Jones, Capers, Quantifying Software: Global and Industry Perspectives; CRC Press, 2017.

Jones, Capers, A Guide to Software Measures and Metrics; CRC Press, 2017.

Jones, Capers, The Technical and Social History of Software Engineering; Addison Wesley, 2014.

Jones, Capers and Bonsignour, Olivier, The Economics of Software Quality; Addison Wesley Longman, Boston, MA; ISBN 10: 0-13-258220—1; 2011; 585 pages.

Jones, Capers, Software Engineering Best Practices; McGraw Hill, New York, NY; ISBN 978-0-07-162161-8; 2010; 660 pages. (Being translated into Chinese by IBM China.)

Jones, Capers, Applied Software Measurement; McGraw Hill, New York, NY; ISBN 978-0-07-150244-3; 2008; 662 pages. (Available in English, Japanese, and Chinese editions)

Jones, Capers, Estimating Software Costs; McGraw Hill, New York, NY; 2007; ISBN-13: 978-0-07-148300-1. (Available in English and Japanese editions)

Jones, Capers, Software Assessments, Benchmarks, and Best Practices; Addison Wesley Longman, Boston, MA; ISBN 0-201-48542-7; 2000; 657 pages.

Jones, Capers, Software Quality: Analysis and Guidelines for Success, 1997.

B. Books by additional authors

Boehm, Barry Dr., Software Engineering Economics; Prentice Hall, Englewood Cliffs, NJ; 1981; 900 pages.

Booch, Grady, Object Solutions: Managing the Object-Oriented Project; Addison Wesley, Reading, MA; 1995.

Capability Maturity Model Integration, Version 1.1, Software Engineering Institute; Carnegie-Mellon Univ.; Pittsburgh, PA; March 2003; http://www.sei.cmu.edu/cmmi/

Brooks, Fred, The Mythical Man-Month, Addison-Wesley, Reading, Mass., 1974, rev. 1995.

Charette, Bob, Software Engineering Risk Analysis and Management; McGraw Hill, New York, NY; 1989.

Charette, Bob, Application Strategies for Risk Management; McGraw Hill, New York, NY; 1990.

Cohn, Mike, Agile Estimating and Planning; Prentice Hall PTR, Englewood Cliffs, NJ; 2005; ISBN 0131479415.

DeMarco, Tom; Controlling Software Projects; Yourdon Press, New York; 1982; ISBN 0-917072-32-4; 284 pages.

Ewusi-Mensah, Kweku, Software Development Failures; MIT Press, Cambridge, MA; 2003; ISBN 0-26205072-2276 pages.

Gack, Gary, Managing the Black Hole – The Executives Guide to Project Risk; The Business Expert Publisher; Thomson, GA; 2010; ISBSG10: 1-935602-01-2.

Galorath, Dan, Software Sizing, Estimating, and Risk Management: When Performance is Measured Performance Improves; Auerbach Publishing, Philadelphia; 2006; ISBN 10: 0849335930; 576 pages.

Garmus, David & Herron, David, Measuring the Software Process: A Practical Guide to Functional Measurement; Prentice Hall, Englewood Cliffs, NJ; 1995.

Garmus, David and Herron, David, Function Point Analysis – Measurement Practices for Successful Software Projects; Addison Wesley Longman, Boston, MA; 2001; ISBN 0-201-69944-3;363 pages.

Glass, R.L., Software Runaways: Lessons Learned from Massive Software Project Failures; Prentice Hall, Englewood Cliffs; 1998.

Hill, Peter R., Practical Software Project Estimation; McGraw Hill, 2010

Harris, Michael, David Herron, and Stacia Iwanicki, The Business Value of IT: Managing Risks, Optimizing Performance, and Measuring Results; CRC Press (Auerbach), Boca Raton, FL: ISBN 13: 978-1-4200-6474-2; 2008; 266 pages.

Watts, Humphrey, Managing the Software Process; Addison Wesley, Reading, MA; 1989.

IFPUG Counting Practices Manual, Release 4, International Function Point Users Group, Westerville, OH; April 1995; 83 pages.

International Function Point Users Group (IFPUG), IT Measurement – Practical Advice from the Experts; Addison Wesley Longman, Boston, MA; 2002; ISBN 0-201-74158-X; 759 pages.

Park, Robert E. et al, Software Cost and Schedule Estimating - A Process Improvement Initiative; Technical Report CMU/SEI 94-SR-03; Software Engineering Institute, Pittsburgh, PA; May 1994.

Park, Robert E. et al, Checklists and Criteria for Evaluating the Costs and Schedule Estimating Capabilities of Software Organizations; Technical Report CMU/SEI 95-SR-005; Software Engineering Institute, Pittsburgh, PA; January 1995.

McConnell, Steve, Software Estimating: Demystifying the Black Art; Microsoft Press, Redmund, WA; 2006.

Roetzheim, William H. and Beasley, Reyna A., Best Practices in Software Cost and Schedule Estimation; Prentice Hall PTR, Saddle River, NJ; 1998.

Strassmann, Paul, Information Productivity; Information Economics Press, Stamford, Ct; 1999.

Strassmann, Paul, Information Payoff; Information Economics Press, Stamford, Ct; 1985.

Strassmann, Paul, Governance of Information Management: The Concept of an Information Constitution; 2nd edition; (eBook); Information Economics Press, Stamford, Ct; 2004.

Strassmann, Paul, The Squandered Computer; Information Economics Press, Stamford, CT; 1997.

Stukes, Sherry, Deshoretz, Jason, Apgar, Henry, and Macias, Ilona, Air Force Cost Analysis Agency Software Estimating Model Analysis; TR-9545/008-2; Contract F04701-95-D-0003, Task 008; Management Consulting & Research, Inc.; Thousand Oaks, CA 91362; September 30 1996.

Wellman, Frank, Software Costing: An Objective Approach to Estimating and Controlling the Cost of Computer Software, Prentice Hall, Englewood Cliffs, NJ, ISBN 0-138184364, 1992.

Whitehead, Richard, Leading a Development Team; Addison Wesley, Boston, MA; 2001; ISBN 10: 0201675267; 368 pages.

Yourdon, Ed, Death March - The Complete Software Developer's Guide to Surviving "Mission Impossible" Projects; Prentice Hall PTR, Upper Saddle River, NJ; ISBN 0-13-748310-4; 1997; 218 pages.

Yourdon, Ed, Outsource: Competing in the Global Productivity Race; Prentice Hall PTR, Upper Saddle River, NJ; ISBN 0-13-147571-1; 2005; 251 pages.

C. Readings on Software Maintenance

Arnold, Robert S., Software Reengineering; IEEE Computer Society Press, Los Alamitos, CA; 1993; ISBN 0-8186-3272-0; 600 pages.

Arthur, Lowell Jay, Software Evolution - The Software Maintenance Challenge; John Wiley & Sons, New York; 1988; ISBN 0-471-62871-9; 254 pages.

Gallagher, R.S., Effective Customer Support; International Thomson Computer Press, Boston, MA; 1997; ISBN 1-85032-209-0; 480 pages.

Grubb, Penny and Armstrong, Takang; Software Maintenance - Concepts and Practice; World Scientific Pub. Co; 2003; ISBN 981-238-425-1.

Kan, Stephen H., Metrics and Models in Software Quality Engineering; Addison Wesley, Reading, MA; ISBN 0-201-72915-6; 2003; 528 pages.

McCabe, Thomas J., "A Complexity Measure"; IEEE Transactions on Software Engineering; December 1976; pp. 308-320.

Muller, Monika and Alain Abram (editors), Metrics in Software Evolution; R. Oldenbourg Vertag GmbH, Munich; ISBN 3-486-23589-3; 1995.

Parikh, Girish, Handbook of Software Maintenance; John Wiley & Sons, New York; 1986; ISBN 0-471-82813-0; 421 pages.

Pigoski, Thomas M., Practical Software Maintenance - Best Practices for Managing Your Software Investment; IEEE Computer Society Press, Los Alamitos, CA; 1997; ISBN 0-471-17001-1; 400 pages.

Polo, Macario et al, Advances in Software Maintenance, Management, Technologies, and Solutions; World Scientific Pub. Co.; 1996; ISBN 981-022826-0.

Sharon, David, Managing Systems in Transition - A Pragmatic View of Reengineering Methods; International Thomson Computer Press, Boston, MA; 1996; ISBN 1-85032-194-9; 300 pages.

Takang, Armstrong and Penny Grubh, Software Maintenance Concepts and Practice; International Thomson Computer Press, Boston, MA; 1997; ISBN 1-85032-192-2; 256 pages.

ARTICLE

3.2 The Changing Face of Systems Engineering Education in the UK

by

Professor Emma Sparks

Email: e.sparks@cranfield.ac.uk

and

Dr. Fanny Camelia

Email: fanny.camelia@cranfield.ac.uk

Cranfield University

Abstract

Our concept of higher education is changing, whether it is the shift in perception to students as customers (Guilbault, 2018) or the desire to extend the concept of apprenticeships to master's level and beyond (Government of the UK, 2015). For Systems Engineering (SE), this presents opportunities and challenges, due to the holistic basis of the discipline and the widening demographic of the learners. How do we meet the requirements of the professional bodies and our industry partners when evaluation of competency is the basis for recruitment and retention, while creating an engaging and challenging learning environment?

STEM (Science, Technology, Engineering and Math) subjects are facing shortfalls in uptake (Lazarus, 2017), compounded by an ageing demographic in the engineering sector potentially leading to a cliff edge of skilled professionals in the next ten years (HR Director, 2017). The question is: how to attract a new generation of systems engineers while providing them with a learning environment that prepares them for the challenges they will face within the workplace?

Master's level apprenticeships may be a way of achieving these outcomes. Apprenticeships are a model of learning that combines work and study to develop the necessary skills and knowledge to meet the specific needs of industry (Which, 2018). Although traditionally adopted for skill development of young people via 'master craftsmen' (Unwin, 2009) the concept has evolved into an education model combining

academic studies and practice for gaining both skills and formal academic qualifications. With a combination of vocational training 'on the job' coupled with professional mentoring and academic studies, the curricula are designed to be 'experiential', fusing knowledge and ability to use that knowledge with emotions, feelings, and perceived value of knowledge and skills (Camelia, 2016). This helps to bridge the gap between education and competency, fulfilling the needs of both the learner and their potential employers.

This paper presents some of the changes and the challenges for Systems Engineering education, primarily at the postgraduate level. The emphasis will be primarily on the UK, but it also addresses the wider initiatives and opportunities internationally.

Introduction

Systems Engineering (SE) is distinct within the wider field of engineering due to its interdisciplinary approach to solving complex problems (Camelia and Ferris, 2016) with implementation believed to have significant impact on financial, schedule, and performance value (Stevens and Brook, 1998; Honour, 2004, 2013; Squires, 2011).

With ever increasing system complexity and increased awareness of the value of SE application in system development projects, the overall demand for systems engineers has increased (Pyster *et al.*, 2015), which in turn has placed greater emphasis on SE education particularly at the post graduate level. The distinctiveness of the discipline creates divergence from the characteristics of most specialist engineering education programs as summarized below (Sage and Armstrong, 2000; Kossiakoff *et al.*, 2011; Muller and Bonnema, 2013).

Specialist Engineering Education Programs	**Systems Engineering Education Programs**
a limited scope	a broader scope
internally focused on specific methods to produce a useful product or service	internally and externally focused on the interactions and interplay between the systems and environment
well-defined problems	ill-defined, incomplete, and ambiguous problems
quantitative relationships	quantitative and qualitative relationships
one correct answer for many engineering problems	no single best answer for many problems

This distinctive holistic nature of the discipline creates both opportunities and challenges, from the diverse learner demographic that we increasingly see, to balancing the needs of the learner with the prospective employers and also the professional bodies that accredit educational programs. These latter stakeholders are looking for something more than just education - they want to evaluate competency, which requires application in context.

There is recognition that a gap exists between education and competence of the learner, with the former emphasizing knowledge and skills and the latter including traits, motives, and self-concept (Walther & Radcliffe, 2007). If education is intended to prepare the learner for professional practice (Lemaitre, 2006), then we may need to expand and revise our educational offerings within SE.

As a result, master's level apprenticeships have come into being within the UK (Government of the UK, 2015). These are employer-led initiatives intended to develop the necessary knowledge, skills, and behaviors to meet the specific needs of industry (Which, 2018), fusing educational and vocational components. With the vocational component being context specific to the employer and mapping to potentially diverse roles and career paths, it overcomes some of the challenges presented in an educational setting of what is 'possible' to teach within a relatively standardized curriculum (Walther & Radcliffe, 2007).

With master's level apprenticeships having been in existence for just over three years in the UK, the higher education community is still trying to grapple with an emerging market and need.

The Need for Experiential Learning Environment in SE Education

The nature of SE as a discipline benefits from practitioners having hands on experience and learning (Williams & Derro, 2008) in order to perform in leading roles (Devgan, Shetty and Zein-Sabatto, 2010). Therefore, SE students need to be equipped with experience of SE activities and methods during their studies to enable them to become professional system engineers capable of dealing with large scale complex problems that require integrated solutions throughout the system life cycle (Sage, 2000).

It may be no coincidence that many of the educational courses in SE are offered at post graduate level with learners coming from a number of disciplines into Systems Engineering. That is not to say that there are no undergraduate SE courses, but a review of educational programs in the US (Squires, 2011) supported a trend of students focusing on a specialized engineering disciplines earlier in their undergraduate studies, followed by a more intensive focus on system-centric SE studies in post graduate programs (Squires, 2011).

A study of the development of student Systems Thinking (ST) (Camelia, 2016), highly regarded as one of the most important competencies in SE, within SE education, found that age and experience are positively associated with student ST development both in the cognitive domain, (related to knowledge and ability to use that knowledge), and the affective domain (related to emotions, feelings, and perceived value of knowledge and skills). In other words, the older the students and the more work experience they have, the higher their cognitive ability in ST and the stronger their emotion, feeling, and value placed on the

importance of the application of ST. The results indicate that age and work experience equip students within experiential learning environments and imply that experiential learning environments are important for developing student cognitive performance and student inclination to the application of SE. This implication is in line with other studies that indicate varied educational opportunities that move away from traditional lecture-based delivery support students in dynamic real-world project environments in engineering education generally (Mills and Treagust, 2003; Al-Bahi, *et al*, 2011), and in the SE education particularly. (Turnquist *et al*., 2000) and (Gonçalves, 2008) suggest that educational interventions need to be adapted. A potential delivery model to address this need is the introduction of master's level apprenticeships.

Apprenticeships are a model of learning that combines work and study with the objective of developing the necessary skills and knowledge to meet the specific needs of industry (Which, 2018). This model was originally adopted for skill development of young people by learning and engaging directly with 'master craftsmen'; most involved payment by parents for the master craftsmen to train their children prior to industrial revolution (Unwin, 2009). The concept has evolved into an education model combining academic studies and practice for gaining both skills and formal academic qualifications. The Systems Engineering Master's Apprenticeship was established in 2016 as part of the Department for Business, Innovation, and Skills Apprenticeship Trailblazer program, and was developed under the sponsorship of the Defense Growth Partnership and Employer Group led by Atkins (Defense Growth Partnership (DGP, 2016). The central government standard and assessment plan created a competency profile linked to job roles that learners can map against. This provides flexibility and specificity for the industry partners involved, with master's level education provided by a number of universities, and vocational experience and mentoring provided by the learner's employer. At the end of the program, learners are expected to demonstrate SE competencies, cognitively and affectively, at the level of INCOSE practitioner.

Having an industry-led initiative has many benefits for a professional discipline. It helps to align industry needs with education and training. Some countries such the UK, Australia, and the United States (US), have and continue to experience a shortage of skilled workers in Science, Technology, Engineering and Math (STEM) and in SE. Despite many initiatives to resolve this issue, STEM and SE positions are not easily filled (Lazarus, 2017). This suggests that the quality of graduates does not meet the required employability competencies required by industry (Walther and Radcliffe, 2007; Lazarus, 2017). This also implies that despite intentions across the engineering disciplines to prepare individuals for professional practice (Lemaitre et al, 2006); educational institutions have not been able to fully prepare their students to meet these needs (Walther & Radcliffe, 2007).

The Need for an Integrated Cognitive-Affective Teaching and Learning Model

The key concerns within industry are associated with 'soft skills' within the criteria of the Accreditation Board for Engineering and Technology (ABET) and the Washington Accord, the international standards for programs in engineering, and within the criteria of some competency frameworks developed by employers and professional societies in SE such as NASA's SE competency model (Academy of

Program/Project & Engineering Leadership, 2009), MITRE competency model (MITRE Human Resources, 2010), INCOSE UK WG (Systems Engineering Competencies Working Group INCOSE-UK, 2010) and ENG competency model (Department of Defense, 2013).

These soft skills are easily associated with the affective domain because of their relationship with the personal values and practices of the students (Ferris, Squires, and Camelia, 2015). However, the affective domain is not conventionally a focus of engineering education and SE programs (Ferris, 2011; Alias *et al.*, 2014; Pyster *et al.*, 2015). One reason is that soft skills are considered to be observable only through long-term evidence while short-term length of courses or programs are deemed inadequate for its assessment (Shephard 2010); thus, teaching and learning in higher education only focuses on developing observable knowledge and skills (Walther and Radcliffe, 2007). Another reason is due to the prevalence of reductionist philosophy and practice in which the separation of 'mind' and 'matter' causes the emotional aspects to be neglected as it is considered 'biased' in relation to the 'normal' cognitive function (Zhang and Lu, 2009). In fact, affective domain development is just as important as cognitive development in SE education, so that students not only know what to do and how to perform a particular SE knowledge and skill, but they also appreciate the importance of that knowledge and skill so they can naturally, routinely, and intuitively use the knowledge and skills when needed, even under pressure of time, budget, or company culture (Camelia and Ferris, 2018).

An integrated cognitive–affective teaching model is needed to design effective teaching and learning environments. Graduate Reference Curriculum for Systems Engineering (GRCSE) has provided generic SE competency outcomes that incorporate the affective domain for master's qualifications (Pyster *et al.*, 2015). However, discussions on how to optimize student achievement of these outcomes are very limited, especially since the GRCSE is intended to have sufficient flexibility that can be tailored by specific institutions across different countries (Henry *et al.*, 2013).

It is believed that providing an experiential learning environment is the answer to the integration of a cognitive and affective teaching model in SE education. Through an experiential learning environment characterized by high levels of participation, learners can take personal responsibility to participate cognitively and affectively to develop knowledge, skills, and attitudes (Gentry, 1990) meeting the wider needs of prospective employers.

As educators, we increasingly need to balance the needs of the learner with that of potential employers and also professional accrediting bodies. Whether or not that is the role of academic institutions is another question, but conversely without providing educational offerings that meet the needs of the market, courses will cease to exist and the shortfall in skilled professionals will potentially continue to rise.

Summary and Conclusion

Systems engineering is a distinct discipline within general engineering disciplines. The unique characteristics of this discipline lead to the emergence of specific requirements for implementing SE education in contrast to other specialized engineering. This article emphasizes the importance of an

experiential learning environment for developing systems engineers and the need for integration of a cognitive and affective teaching model in SE education. It is believed that apprenticeships at master's level within SE is an appropriate model to fulfil the requirements for skilled systems engineers.

List of Acronyms Used in this Paper

Acronym	Explanation
ABET	Accreditation Board for Engineering and Technology
BKCASE™	Body of Knowledge and Curriculum to Advance Systems Engineering
DGP	Defense Growth Partnership
DoD	Department of Defense
GRCSE	Graduate Reference Curriculum for Systems Engineering
INCOSE	International Council on Systems Engineering
SE	Systems Engineering
SEBoK™	Systems Engineering Body of Knowledge
ST	Systems Thinking
STEM	Science, Technology, Engineering, and Math
UK	United Kingdom
US	United States

References

Academy of Program/Project & Engineering Leadership 2009. *NASA's systems engineering competencies.* Washington, D.C.: U.S. National Aeronautics and Space Association [Online]. Available:

https://www.nasa.gov/offices/oce/divisions/appel. (Accessed 31st July 2018).

Al-Bahi, A. M., Abdulaal, R. M. S., Soliman, A. Y. and Iskanderani, F. I. (2011) "Introductory project-based design course to meet socioeconomic challenges", in *Annual Conference and Exposition*, Vancouver, BC.

Alias, M. et al. (2014) "Translating theory into practice: Integrating the affective and cognitive learning dimensions for effective instruction in engineering education", *European Journal of Engineering Education*, 39(2), pp. 212–232. Doi: 10.1080/03043797.2013.838543.

Brown, D. E. and Scherer, W. T. (2000) "A Comparison of Systems Engineering Programs in the United States", *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 30(2), pp. 204–212.

Camelia, F. (2016) *Systems thinking in systems engineering education*. University of South Australia.

Camelia, F. and Ferris, T. (2016) "Systems thinking in systems engineering", in 26th Annual INCOSE International Symposium, p. 18.

Camelia, F. and Ferris, T. L. J. (2018) "Validation studies of a questionnaire developed to measure students' engagement with systems thinking", in *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 48(4), pp. 574–585. Doi: 10.1109/TSMC.2016.2607224.

Camelia, F., and Ferris, T. (2016) "Systems thinking in systems engineering". *26th Annual INCOSE International Symposium (IS2016)*. Edinburgh, Scotland, UK. July 18-21 2016.

Camelia, F., and Ferris, T. (2017) "Undergraduate students' engagement with systems thinking: results of a survey study". *IEEE Transactions on Systems, Man, and Cybernetics: Systems IEEE Early Access Article*. Vol. 47. Issue: 12. pp 3165-3176.

Defense Growth Partnership (DGP) (2016) Systems Engineering Masters Apprenticeship Program.

Lemaitre, Denis, Le Prat, Raymond, De Graaff, Erik and Ludovic Bot (2006) "Editorial: Focusing on competence", *European Journal of Engineering Education*, 31:1, 45-53, DOI: 10.1080/03043790500478390

Devgan, S., Shetty, S. and Zein-Sabatto, S. (2010) "Integrating systems engineering in mainstream engineering disciplines", in *ASEE Annual Conference & Exposition*. Louisville, Kentucky: American Society for Engineering Education.

Department of Defense (2013). SPRDE-SE/PSE competency model, in Defense Acquisition University (DAU)/U.S. Department of Defense.

Ferris, T. L. J. (2011) 'Bloom's affective domain in systems engineering education', in 5th Asia-Pacific Conference on System Engineering (APCOSE). Seoul, Korea.

Ferris, T. L. J., Squires, A. F. and Camelia, F. (2015) 'Integrating affective engagement into systems engineering education paper', in ASEE Annual Conference and Exposition. Seattle, Washington, p. 26.985.1-26.985.17.

Gentry, J. W. (1990) "What is experiential learning", *Guide to business gaming and experiential learning*, 9(20).

Gonçalves, D. (2008) "Developing systems engineers", *Portland International Center for Management of Engineering and Technology* (PICMET), Proceedings, pp. 1963–1972. Doi: 10.1109/PICMET.2008.4599817.

Government of the UK, (2015). https://www.gov.uk/government/publications/higher-and-degree-apprenticeships (Accessed 16th July 2018).

Guilbault, M. (2018) "Students as customers in higher education: The (controversial) debate needs to end." *Journal of Retailing and Consumer Services*. Vol. 40. Jan 2018, pp. 295-298.

Henry, D., Ferris, T., Squires, A., and Embry-Riddle, M. (2013) "Organizing the graduate reference curriculum for systems engineering (GRCSE) for international relevance". *American Society for Engineering Education International Forum. Omni Hotel at CNN Center, Atlanta*. Saturday 22nd June 2013.

Devanandham, Henry; Ferris, Tim; Squires, Alice F; Towhidnejad, M. (2013) "Organizing the Graduate Reference Curriculum for Systems Engineering (GRCSE) for International Relevance", in ASEE International Forum.

Honour, E. C. (2004) "Understanding the value of systems engineering", in INCOSE International Symposium. Toulouse, France, pp. 1–16.

Honour, E. C. (2013) *Systems Engineering Return on Investment*. University of South Australia.

HR Director (2017). https://www.thehrdirector.com/features/career-development/live-and-learn/ (accessed 16th July 2018).

Kolb, D. A. (2014) *Experiential learning: Experience as the source of learning and development*. FT press.

Kossiakoff, A. et al. (2011) *Systems Engineering Principles and Practice.* 2nd Ed. New Jersey: Wiley-Interscience.

Lazarus, D. (2017) Australian engineering student and alumni perceptions of self-responsibility in employability competencies development. University of South Australia.

Lemaitre, D., Prat, R. L., et al. (2006). “Editorial focusing on competence”. *European Journal of Engineering Education*. Vol. 31, No. 1, pp45-53.

Mills, J. E. & Treagust, D. F. (2003) “Engineering education - Is problem-based or project-based learning the answer?”, *Australian Journal of Engineering Education*.

Mitre Human Resources (2010). MITRE systems engineering (SE) competency model. [Online]. Available: https://www.mitre.org/sites/default/files/publications/10_0678_presentation.pdf, (accessed 31st July 2018).

Muller, G. and Bonnema, G. M. (2013) “Teaching Systems Engineering to Undergraduates; Experiences and Considerations”, INCOSE International Symposium, 23(1), pp. 98–111. Doi: 10.1002/j.2334-5837.2013.tb03006.x.

Pyster, A. et al. (2015) “Graduate Reference Curriculum for Systems Engineering (GRCSE)”, *Guide to the Systems Engineering Body of Knowledge (SEBoK*). Available at: www.bkcase.org/grcse/.

Sage, A. (2000) “Systems Engineering Education”, *IEEE Transactions on Systems, Man, and Cybernetic-Part C: Applications and Reviews*, 30(2), pp. 164–174.

Sage, A. P. and J.E. Armstrong, J. (2000) *Introduction to Systems Engineering*. New York: John Wiley and Sons, Inc.

Shephard, K. 2010. "Higher education's role in 'education for sustainability". Australian universities review, 52, pp 13-22.

Squires, A. F. (2011) “Investigating the relationship between online pedagogy and student perceived learning of systems engineering competencies”. Available at: /Users/Fernando/Documents/Project Based learning/phdthesis education/Squires2011.pdf

Stevens, R. and Brook, P. (1998) *Systems Engineering: Coping with Complexity*. Harlow, Essex: Prentice Hall. Doi: 10.1049/em:19980613.

Systems Engineering Competencies Working Group INCOSE-UK (2010). Systems Engineering Competencies Framework, San Diego, California: International Council of Systems Engineering (INCOSE).

Turnquist, M. A. et al. (2000) “Designing a systems engineering program using academic/industry collaboration”, INCOSE 2000, pp. 799–805.

Unwin, A. F. & L. (2009) “Change and continuity in apprenticeship: the resilience of a model of learning”, *Journal of Education and Work*, 22(5), pp. 405–416.

Walther, J., and Radcliffe, D. (2007) “The competence dilemma in engineering education: Moving beyond simple graduate attribute mapping”. *Australian Journal of Engineering Education*. Vol. 13, No.1. pp 41-51.

Walther, J., Kellam, N., Sochacka, N., and Radcliffe, D. (2011) “Engineering competence? An interpretive investigation of engineering students’ professional formation”. *Journal of Engineering Education*. Vol. 100. No. 4. pp 703-740.

Which.co.uk (2018). The complete guide to higher and degree apprenticeships. https://university.which.co.uk/teachers/introduce-higher-education-options/higher-and-degree-apprenticeships-guide-download (accessed 23rd July 2018).

Williams, C. & Derro, M. E. (2008) *NASA System Engineering Behavior Study*.

Zhang, W. and Lu, J. (2009) “The practice of affective teaching: A view from brain science”, *International Journal of Psychological Studies*, 1, pp. 35–41.

About the Authors

Professor Sparks worked for a number of years as a research scientist within government before transitioning to academia in 2005. She is the Head of the Centre for Systems Engineering at Cranfield University, responsible for a team delivering post graduate education, research, and consultancy. A Senior Fellow of the Higher Education Academy, Professor Sparks has spent a number of years designing and delivering post graduate educational programs in Systems Engineering with her Centre developing the first-ever Level 7 apprenticeship in Systems Engineering, in response to a UK government initiative. A 2018 winner of the Top 50 women in engineering and a member of INCOSE, Professor Sparks is passionate concerning promoting STEM skills (Science, Technology, Engineering and Math) and for others to recognize the benefits of applying systems thinking and philosophy to understanding complex problems across domains.

Dr. Camelia holds a B.Eng. in Industrial Engineering from Andalas University, Indonesia, M.PrjMgmt. in Applied Project Management (Defence) from University of Adelaide, Adelaide, South Australia and Ph.D. in Systems Engineering from University of South Australia, Mawson Lakes, Australia. She is a Senior Research Fellow at the Centre for Systems Engineering, Cranfield University, at Shrivenham, UK. Her research interests include systems thinking, SE education, dynamic simulation and modelling, model-centric SE, systems of system approach, and SE/project management crossover.

ARTICLE

3.3 Integrating Program Management and Systems Engineering

by

Ralph R. Young

Editor, SyEN

Email: ryoung@ppi-int.com

This month we provide a summary of Chapter 14, Successful Change Programs that Improved Integration, in *Integrating Program Management and Systems Engineering* (IPMSE), a collaboration of the International Council on Systems Engineering (INCOSE), the Project Management Institute (PMI), and the Consortium for Engineering Program Excellence (CEPE) at the Massachusetts (USA) Institute of Technology (MIT). This is our fifteenth article in this series. Our objective in providing this series is to encourage subscribers to leverage the research base of this book that has provided new knowledge and valuable insights that will serve to strengthen the performance of complex programs. “The Book” is highly

recommended as professional development for all systems engineers and is available to members of INCOSE at a discount.

Chapter 14 is the second of three chapters in Part III of The Book, labelled Developing Integration Competencies in Your Organization (Chapter 13 is titled Integration Means Change; Chapter 15 is titled Leading an Integration Change Program).

Chapter 14 provides five case studies describing change efforts to develop or strengthen integration capabilities in engineering programs and organizations. They illustrate different approaches taken to create the changes needed to increase the level of integration between program management and systems engineering. The case studies illustrate the elements of the Integration Framework developed in Chapter 6 and provide examples of ways to implement them. It was found that each organization and engineering challenge is different, and that meaningful change in processes, practices, and mindsets comes from active and sustained efforts by the organization and its leaders.

In the first case study, involving integration of program management and systems engineering at Lockheed Missiles & Space Company (LMSC) in 1957, an integrated management approach was developed in the years when the systems engineering discipline was still in its early stages. The challenge in this example was the repeated technical failure of complex systems in a new and challenging technology area. The need for change was clear and urgent with each successive and costly mission failure and the inability of the program to deliver any tangible benefits. The implementation of a rigorous technical process by defining and integrating the chief systems engineer into the program organization structure provided essential information to the program management to improve the quality of decisions and coordination across the entire program. The program manager was therefore better able to understand and influence technical issues, including those at the subcontractor level. Mission success rates improved dramatically, and the organization demonstrated sustained change by continued use of the practices by diffusing them into follow-on programs and through their impact on the development of the work force. Progression from chief systems engineer to program manager became the norm for the LMSC Space Systems Division organization and several presidents of that Division were appointed through such a chain of assignments. Systems engineering provided the “layer” necessary to connect the highly complex engineering reality with the requirements of managing a large program.

The actions taken in this example to increase integration included:

- Integrating planning and resource allocation for engineering efforts into the program manager’s responsibilities;
- Restructuring roles of managers and functional leads in the program and in the division;
- Developing policies to manage staffing to ensure an appropriate and equitable balance of the right technical people across the organization;

- Developing leadership transition paths from chief systems engineer through program manager and above;
- Adapting to challenges by testing new integration approaches over time to see what works; and
- Pilot testing new ideas on one program, and then diffusing best practices to others.

CIA employees increasingly appreciated the relevance of systems engineering in their projects; indeed, 89% of them reported that it was applicable to their jobs, a full 30 points higher than they reported prior to their certification program.

During the late 2000s, the Federal Acquisition Institute developed and implemented the Federal Acquisition Certification for Program and Project Managers (FAC-P/PM). That certification was created to extend the integration of project management and systems engineering from the U.S. Department of Defense (DoD) to all civilian federal employees, especially those involved with limited scope and lower complexity projects. The approach called for creating a competency-based program and included structuring the program in three levels; some systems engineering was included in all levels. The competencies that underpinned this program were developed based on surveys of subject matter experts at many of the federal agencies. These surveys, which were based on DoD Directive 5000 ("The Defense Acquisition System"), identified how often and how important particular project management and systems engineering activities were. The competencies resulting from these surveys were mandated by the Office of Federal Procurement Policy in 2007.

The Department of Veterans Affairs (VA) was one of the agencies that was out in front with the implementation of FAC-P/PM discussed above. In 2008, the VA formed an Acquisition Academy to address the growing acquisition workforce challenges they were facing; a program management school was created within that academy that tailored the FAC-P/PM to the VA mission and work force. It too integrated project management and systems engineering into a competency-based program, was targeted at those involved with the smaller and less complex projects, and was structured in three levels with some systems engineering at each level. Post-training metrics indicated that students returned to their jobs able to successfully complete projects they otherwise would have struggled with or failed.

The actions taken in this example to increase integration included:

- Create integrated certification for program managers that drew on both program management and systems engineering standards to raise program manager awareness of systems engineering.
- Provide training to project and program managers on a widespread basis, regardless of the size of their projects or programs.
- Provide multi-tiered certification system to help increase the depth of understanding about the integration issues between program management and systems engineering.
- Capture successful practices and insights and disseminate them across multiple locations.

Editor's note: It seems that many complex projects and programs could implement the above actions effectively and efficiently at low cost. Given the research base and results of this book, it seems that serious consideration should be given to doing this.

The third case study involved integrated software and program management at the Nationwide Mutual Insurance Company, one of the largest and most diversified financial services companies in the United States. The company provides products such as retirement planning, auto and home insurance, farm owner insurance, and commercial lines insurance to end consumers and intermediaries. Insight at Nationwide was strongly driven by the adoption within senior IT executives of a "growth" versus a "fixed" mindset, inspired by Carol Dweck's (2006) book, *Mindset: The New Psychology of Success*. Traditional program management focuses on planning, change control, and periodic status reporting. Agile software development focuses on discovery, short learning cycles, visual management, and daily huddles. These two perspectives are set up to be in conflict with each other. To correct these differences, the company undertook a transformation in how software was developed, and the roles and expectations of all those involved with the process. A new organization was created, the Application Development Center (ADC), with a focus on technical excellence and process discipline.

The ADC built upon the successes of the original agile teams, while incorporating the supporting roles, including the program manager, directly into the model. Standardized role definitions and behaviors, referred to as "standard work" were defined for everyone. Accountability and reinforcement systems were put in place. All of these things were put in place. The deliberate intention was to put in place a program that learns and adapts, continuously improving process and practices.

A specific software development method, Agile, was utilized, but with outcomes that ranged from significant success to disappointment. The cause of the variation in success was determined to be a misalignment between the program management organization and the technical functions. The company embarked on a number of change initiatives managed through an existing Lean Six Sigma improvement program infrastructure in the organization to transform how software was developed, including the roles and expectations of all those involved in the process, a focus on technical excellence and process discipline including definition of "standard work", the development of a new mindset concerning collaboration and interaction, and the application of new metrics to evaluate performance. Not only did quality and productivity improve significantly, but also the engagement of the employees and the atmosphere of collaboration between program management and technical functions was markedly improved and sustained as a result.

The actions taken in this example to increase integration included:

- A change program to improve management integration with developers, leveraging existing change agents and infrastructure to roll out change initiatives across the company.
- Tracking down the root cause of inconsistent execution of agile across the organization: a lack of process discipline and an attitude of exclusion between program and project management and the software developers.

- Redefining the program management orientation to better align with the agile development approach.
- Creating the Application Development Center with a focus on technical excellence and process discipline.
- Creating standard work for managers explaining how they should interact with the development teams, including reporting and priorities.
- Developing a new set of performance metrics for program managers to track development progress.

The fourth case study involved boosting productivity in BMW's engineering department. A division-wide formal change program was implemented that involved a number of change projects. Top-down, large scale transformation projects tackled strategic, central themes. Bottom-up initiatives provided processes and method for enhancing value creation by individuals and teams. An outreach program encouraged participation by each member of the organization to achieve a new mindset focused on collaboration. Engineering productivity improved dramatically, and importantly the company avoided disruption during the global economic crisis, expanded its product line with no additional resources required, and was able to achieve its strategic objectives.

This case study describes a change program that unfolded over the period from 2007 to 2012. It was managed by a program management office and involved a large number of individual projects to bring about the overall vision of the BMW Group. The integration of the many projects, most of them technical, resulted in a more integrated program management and engineering environment. The Engineering Division created an engineering transformation program titled the E3 Program that had five major aspects. The E3 Program was put into place to transform the engineering organization to a state in which it would be able to support the very ambitious corporate strategy including an expansion of the model range, increase in vehicle quality, and reduction of engineering cost per vehicle, given a stable number of employees. The transformation activities addressed three top-level Challenges, expressed in the name "E3":

- **Exhilarating products.** How could the engineering division be closer to the customers' needs? How could engineering activities better focus on what really creates value for the customer, and ultimately increase customer satisfaction and grow the customer base? How could the company change some fundamental attitudes concerning "what the customer wants", and how could the company technically deliver through its products?
- **Efficient processes and structures.** How could the company develop better products, in less time, and for less money? How could the company develop more products, and more exciting products, and improve productivity and innovation without driving up cost and lead time?
- **Emotions and team spirit.** How could the company develop openness and willingness for change? How could the company assure that employees take ownership of and responsibility for

change? How could the company enable managers to be effective leaders of change? How could the company maintain an attractive and inspiring work environment, as well as keep everyone employed while at the same time drastically improving productivity?

The actions taken in this example to increase integration included:

- Senior leaders provided the vision and resources for the change and were personally involved in supporting it.
- The company and division strategy were communicated to all employees in special meetings and used to define the change projects.
- A PMO managed the change program and provided resources to enable local leaders and employees to implement the change. An internal professional change management group with change agents, toolsets, and a model for change assisted the PMO.
- Projects included both top-down and bottom-up change initiatives to engage the entire workforce.
- The scope of changes addressed product, processes, tools, communications, leadership, and culture.

Editor's note: Dealing with change is a constant requirement in today's business environment. Business processes must incorporate strategies to deal with change successfully. To that end, the actions noted above might be considered by leaders involved in all complex projects and programs. It is apparent from this example (among others) that change management is a critical driver in obtaining successful business results. (Note that the previous issue of PPI SyEN (PPI SyEN 68, August 2018) included summaries of several change management resources in the SE Publications section.)

The final case study presented in Chapter 14 of The Book involves delivering the world's most complex inner-city infrastructure program, Boston's "Big Dig". Almost 30 years in the making, the front-end planning phase began in the 1970's, the first shovels went into the ground in the early 1990's, and the program was substantially completed in 2007. The program was complicated by the fact that it was staged as a fast-track program where initiation was beginning on some projects while closure had been achieved on other parts of the program.

The Big Dig, like most large-scale infrastructure programs, grew from a vision of a small group of people who saw a city in need of revitalization. The program had numerous challenges, including working in one of the most congested urban areas in the country, and coordinating more than 132 major work projects, 54 major design packages, thousands of subcontractors, more than 9,000 processes and procedures, and organizing more than 5,000 workers during its peak years of construction. The Big Dig was not always on schedule and budget; however, it did eventually deliver one of the most complex, inter-city tunneling efforts in the world.

Systems engineering was used at the Big Dig to integrate the various components of the 135 major projects in the program and provide a holistic view to the complex engineering systems and requirements. One of the important roles of systems engineering was analysis of the safety failure modes for all critical infrastructure projects.

One of the most important mechanisms for problem solving, conflict resolution, and the reduction of tension among participants on the Big Dig Program was through the Partnering Process.[4] The concept of partnering was first utilized by DuPont Engineering on a large-scale construction project in the mid-1980's, and the U.S. Army Corps of Engineers was the first public agency to use partnering in its construction projects. Partnering is now widely used by numerous government and construction entities around the world.[5] It involves an agreement in principle to share risk and to establish and promote partnership relationships. On the Big Dig Program, partnerships were used to improve schedule adherence, quality, safety, and performance, as well as to reduce risk, costs, claims, disputes, and litigation. Partnering on the Big Dig Program was initially implemented in 1992, primarily on construction projects, but its success in construction later led to its use in design contracts, community groups, and the development of internal and interagency partnerships. Almost 100 partnerships existed on the Big Dig Program, based on contract values ranging from US$4 million to a half billion.

The actions taken in this example to increase integration included:

- Giving leaders shared roles (for example, the engineering manager/deputy program manager) to encourage an integrated perspective.
- Directing project managers and engineers to work toward shared goals with specific direction taking into account the perspectives of other disciplines in making decisions.
- Introducing an IPO and integrated team structures.
- Establishing comprehensive risk management and quality assurance integrated into the component projects.
- Integrating performance reviews that included overall technical compliance with scope and schedule specifications, start-up, testing and test data, and approval activities at several levels prior to the acceptance of and payment for any contract.
- Providing rewards for innovation to the team rather than to individuals.

4 See Partnering in the Construction Industry: A Code of Practice for Strategic Collaborative Working, for an in-depth presentation concerning how to utilize the Partnering Process. Available at https://www.amazon.com/Partnering-Construction-Industry-strategic-collaborative/dp/0750664983/ref=sr_1_2?ie=UTF8&qid=1532203157&sr=8-2&keywords=partnering+in+construction

5 The author recommended use of the Partnering Process when employed by Northrop Grumman Information Technology Defense Systems Group for a program at the U.S. State Department. The facilitator selected to assist with implementing the Partnering Process was Charles Markert. The implementation was highly effective. See www.thepartneringfacilitator.com

- Creating partnering provisions in contracts and having periodic partnering sessions to provide common awareness and commitment across the program.

Summary of Chapter 14: Successful Change Programs that Improved Integration

Five case studies of successful change programs that improved integration were reviewed. They were selected from others reviewed during the five-year research phase of development of the book because they demonstrate a diverse set of challenges and approaches to improving integration and collaboration across management and technical functions. One observes significant differences in the degree of urgency to accomplish the change, the scope of the change effort, and the approach taken. Each case study illustrates deliberate, systemic approach to improving elements of integration in the organization. Some elements of the Integration Framework developed in Chapter 6 of The Book were observed in all five of the case studies. They demonstrate how these elements were applied successfully across a range of organizational and program settings.

In summary, in consideration of the knowledge and insights made available in Chapter 14 of The Book, it is both amazing and terribly unfortunate that many complex projects are not more successful and effective in achieving the results anticipated by stakeholders. *It's not that we don't have the information required to be successful and effective; it's that we don't find and apply the information and knowledge that is available as well as we could.* This concern also (not surprisingly) involves people – how can we get people to work together and support one another? Much knowledge and many insights concerning that which we might do have been provided in this chapter. Some ideas that seem relatively easy to implement and apply have been highlighted above. How many complex projects will take advantage of this information and make investments to further strengthen and improve the application of systems engineering? What do you plan to do as a result of taking the time and effort to read this review of Chapter 14 of *Integrating Program Management and Systems Engineering*? What can we do as the systems engineering community to take a giant step forward in our practice of systems engineering? Who are the leaders who will provide the vision, direction, and motivation to make it happen? How do we garner their attention, commitment, and support for our desperate situation?

Food for Thought

1. Which of the five case studies relates the challenges to integration that you have experienced in your own workplace? What specific elements of the case study are most relevant to your situation?

2. Suppose that you have been requested to consider options for improving the integration of program management and systems engineering in your organization. Select one of the case studies as a model for the challenges that your organization faces. Identify the following:

    a. What is the general situation of the organization and the particular integration challenge you face?

b. What principle(s) of integration would you propose applying to resolve the challenge? What aspects of The Book, *Integrating Program Management and Systems Engineering*, can be used and how might they be deployed?

c. How would you approach the implementation of these integration principles into your organization?

d. What challenges to implementation would you expect to encounter, in the near term and in the longer term?

References

Dweck, Carol. *Mindset: The New Psychology of Success*. New York: Random House, 2006. Available at https://www.amazon.com/Mindset-Psychology-Carol-S-Dweck/dp/0345472322/ref=sr_1_5?s=books&ie=UTF8&qid=1532290660&sr=1-5&keywords=Mindset%3A+The+New+Psychology+of+Success&smid=ATVPDKIKX0DER

Fosberg, Kevin, Mooz, Hal, and Cotterman, Howard. *Visualizing Project Management*. Hoboken, New Jersey USA: John Wiley & Sons, 1996. Available at https://www.amazon.com/Visualizing-Project-Management-Frameworks-Mastering/dp/0471648485/ref=sr_1_1?s=books&ie=UTF8&qid=1532290736&sr=1-1&keywords=Visualizing+Project+Management

Markert, Charles. The Partnering Facilitator. See http://www.thepartneringfacilitator.com/

Tobin, James. *Great Projects: The Epic Story of the /Building of America, from the Taming of the Mississippi to the Invention of the Internet*. New York: Free Press, 2001. Available at https://www.amazon.com/Great-Projects-Building-Mississippi-Invention/dp/0743210646/ref=sr_1_1?s=books&ie=UTF8&qid=1532290793&sr=1-1&keywords=Great+Projects%3A+The+Epic+Story+of+the+%2FBuilding+of+America%2C+from+the+Taming+of+the+Mississippi+to+the+Invention+of+the+Internet

Young, Ralph R. *Project Requirements: A Guide to Best Practices*. Vienna, Virginia USA: Management Concepts, 2006. Available at https://www.amazon.com/Project-Requirements-Guide-Best-Practices/dp/1567261698/ref=sr_1_1?s=books&ie=UTF8&qid=1532290592&sr=1-1&keywords=Project+Requirements%3A+A+Guide+to+Best+Practices

4. SYSTEMS ENGINEERING NEWS

4.1 INCOSE Systems Engineering Competency Framework Released

The INCOSE Competency Framework was launched at the INCOSE International Symposium in July 2018 and is available to members of INCOSE for free download from the INCOSE Store.

What is the INCOSE Competency Framework for Systems Engineering?

The INCOSE Competency Framework provides a set of 36 competencies for Systems Engineering within a tailorable framework that provides guidance for practitioners and stakeholders to identify knowledge, skills, abilities, and behaviors crucial to Systems Engineering effectiveness.

Why was the INCOSE Competency Framework created?

The INCOSE Competency Working Group (CWG) produced the framework to improve the practice of Systems Engineering. The framework along with adoption of effective competency management approaches is intended to be used by customer organizations to produce competency models specifically tailored to their unique needs.

How will the INCOSE Competency Framework help me?

This is a generic framework. It can be applied in the context of any application, project, organization or enterprise for both individual and/or organizational assessment and/or development. The framework is expected to be tailored to suit the application and domain in which it is applied, combining competencies identified herein with others taken from complimentary frameworks (e.g. Program Management, Human Resources, Aerospace, Medical), or generated organizationally, to define the required knowledge, skills and behaviors appropriate to an area or role.

4.2 INCOSE Announces Canada Chapter and Chapter President

David Morris to Lead INCOSE Canada

by

Christine Kowalski

The International Council on Systems Engineering (INCOSE) has introduced INCOSE Canada as the organization's newest chapter. With 70 Chapters overall, INCOSE is dedicated to connecting systems engineering professionals with outstanding networking opportunities while producing state-of-the-art products to discover solutions for today's social and technical problems.

In addition to these goals, INCOSE Canada aims to facilitate the formation of local INCOSE branches throughout Canada and encourage their involvement. As a national Chapter, INCOSE Canada will focus on advocating for the increased professionalism and breadth of use of systems engineering in industry, academia and the federal government.

"By promoting professional development through training and certification opportunities, INCOSE Canada looks forward to educating and connecting systems engineers on an academic and industry-wide level," said David Morris, Chapter President.

Over a career of 38 years, Morris has wide experience in the engineering of land, sea and air projects in the UK Ministry of Defense, industry and Canada's Department of National Defense (DND). First as a naval architect, then a systems engineer and systems engineering manager, Morris oversaw construction,

modernization, and regulations of the government's systems including submarines, SAR aircraft, frigates, and multiple role replenishment ships. He is also a member of the Society of Naval Architects and Engineers.

Morris will be joined by fellow INCOSE Canada officers Vice President Michael Meakin, Secretary Liviu Dancea and Treasurer Philippe Krutchen. INCOSE Canada has selected Daniel Amyot as director of programs, Steven Gibbon as director of communications, and Derya Marquette as director of membership.

4.3 Systems Change: A Growing Need for The Shift to Take Root

by

Benjamin Taylor

Chief Executive, Public Service Transformation Academy

(From the perspective of Benjamin Taylor)

We are about to discover, if we didn't know it already, that local government is all about systems change – impacting the wider system more effectively through a wider range of levers than public service delivery alone can ever achieve.

Yet we are still recruiting and incentivizing local authorities and partner organization chief executives for organizational grip: 'strong management and don't us get on the wrong side of the inspectors'.

Without doubt, it is critical that we maintain high professional standards – some of our services are life-and-death, others mean the difference between losing and gaining vital jobs. But this dynamic – even with a bit of 'partnership working' thrown in – only reinforces the assumptions and silos that prevent real change.

The field of systems change has emerged internationally from a combination of philanthropy, environmentalism, community development, futures studies, and system design. But it's also a natural extension of place shaping, and the best of local government work, from the Wigan Deal to community development in Plymouth – the kind of examples set out in our Public Service: State of Transformation reports.

So, it's about starting from citizen power, focusing on outcomes, and co-creation of public value through a mixed market of funding as well as delivery – the real application of systems thinking.

Systems change seems like a natural end point. As you work with public services, you realize that the power and leverage is and should be in the hands of the citizens. As you work on cross-organization leadership, you realize that holistic thinking is needed.

When you try to really tackle wicked problems, you realize that no intervention will do it: the whole thing has to shift. Systems change is surely coming.

But the reality is, we're stuck in Silo Wars, Episode IV: The Organization Strikes Back.

Our leadership focus is on preventing organizational problems, being good enough at the slicing and dicing, and fiddling the figures to keep the wolves from the door for one more medium-term financial strategy. Our skills (despite an occasional nod to innovation, and an occasional standout leader) are focused on good, old-fashioned organizational control.

Henry Kippin and Matthew Taylor, in their 2017 report for Solace Ignite, quote a senior leader: "What drives too many organizations … is the question: 'What can I be sacked for and how can I avoid it?' Until we can shift this, we will be setting up the pivot point in our public service systems – the leaders of place – to speak one language and act out another.

Ironically, of course, civic municipalism and the beginning of public services arose from precisely the kind of entrepreneurialism that systems chance advocates. So perhaps we can find our way back to a future that allows systems change to really take root.

Article source

4.4 Object Management Group Issues Request for Proposals for A SysML V2 API and Services Standard

The Object Management Group® (OMG®), an international, open membership, not-for-profit technology standards consortium, issued a SysML® v2 API and Services RFP to specify requirements for an Application Programming Interface (API) that includes standard services to create, read, and update SysML v2 models, and connect SysML v2 models with models in other disciplines, such as mechanical, electrical, software, manufacturing, and logistics.

Dr. Manas Bajaj, the SysML v2 API and Services RFP Working Group Chair, said "The SysML v2 API and Services RFP complements the SysML v2 RFP, which was issued last December. A standard API for SysML v2 will significantly enhance the interoperability of SysML modeling tools with each other and with other engineering tools and enterprise services. More enterprise applications and users will be able to produce and consume standard SysML models and participate in the engineering of complex systems."

Both members and non-members of OMG may show their interest in participating in the process by submitting a Letter of Intent by December 10, 2018. In order to become a submitter, individually or as part of a submission team, companies must become members of OMG by the initial submission deadline of February 1, 2020.

About the OMG

The Object Management Group® (OMG®) is an international, open membership, not-for-profit technology standards consortium with representation from government, industry and academia. OMG Task Forces develop enterprise integration standards for a wide range of technologies and an even wider range of

industries. OMG modeling standards enable powerful visual design, execution and maintenance of software and other processes. Visit http://www.omg.org for more information.

4.5 Iris Rivero Named Department Head of Industrial and Systems Engineering at Rochester Institute of Technology

by

Michelle Cometa

Email: macuns@rit.edu

Iris Rivero, a faculty-researcher at Iowa State University and an expert in additive and hybrid manufacturing, has been named department head of RIT's industrial and systems engineering program in the Kate Gleason College of Engineering. She has also been appointed to the Kate Gleason Professorship, a prestigious designation in the engineering college, to support ground-breaking research initiatives in the different engineering disciplines.

"This is an exciting place, and it is already well positioned in these areas with even more potential to expand in complementary areas," said Rivero, who took her position on August 15, "With all the growth in the areas of industrial and systems engineering and additive manufacturing, this new role was attractive from that perspective."

While at Iowa State, Rivero was an associate professor in the university's Industrial and Manufacturing Systems Engineering Department. She led the Interdisciplinary Manufacturing Engineering and Design Laboratory and collaborated on varied projects with partners from academia to national and international corporations. Some of her work included the development of new metal alloys for 3D printing applications and of novel biomaterials that can be used as antibacterial sutures.

Outside of academia, Rivero worked in the engineering departments of Honeywell Engines & Systems and at Advanced Technology Allied Signal Engines & Systems. In 2012, she was named Woman of the Year (Higher Education) by the Hispanic Association of Women and received the Outstanding Young Manufacturing Engineer Award by the Society of Manufacturing Engineers.

Born and raised in Puerto Rico, Rivero credits her father for influencing her decision to seek out engineering as a career. She would go on to receive her bachelor's, master's and doctoral degrees in industrial and manufacturing engineering from Penn State University. She is also eager to continue outreach to under-represented communities to encourage them to pursue STEM programs.

"When I was growing up, engineering was an option that my father taught me about, but many people around me were not thinking about that. And to me, creating programs and communities on campus where they can see people that are like themselves, that is another part of this new role that I am interested in."

Rivero will replace Michael Kuhl, who served as interim department head during the 2017-18 academic year. Kuhl remains with the department as professor and continues research in the area of operations methodologies and next-generation materials handling systems.

The Kate Gleason Professorship is named after Kate Gleason—business leader, inventor, and the first female member of the American Society of Mechanical Engineers—and supported through funding from the Gleason Foundation. The foundation and Gleason family have invested extensively over the years in professorships and scholarships and donated state-of-the-art equipment to the college and RIT. The Kate Gleason College of Engineering is the only engineering college in the U.S. named after a woman.

4.6 Participation Welcome in Shaping SysML ® 2.0

SysML Timeline

- SysML 1.0 was launched in 2006 as a general-purpose graphical modeling language.
- SysML is used in critical systems, from engineering chemical plants to architecting submarine systems.
- Systems engineers, tool vendors, and academia have learned much from this experience.
- The SysML specification has continued to evolve through the OMG Revision Task Force process, however, the scope of the task force limits how much change can be introduced to address the needs.
- It's time for SysML v2.0.

You Can Help Develop SysML v2.0

The Object Management Group® (OMG®) has issued a Request for Proposals for the next version of SysML: SysML v2.0.

- Improve the precision, expressiveness, and usability relative to SysML v1.
- Improve support for MBSE in the broader context of Model-based Engineering (MBE).
- Specify additional features that include model interchange and formal semantics.

How to Participate

- Respond: Organizations interested in submitting a proposal, either alone or as part of a group, must submit a Letter of Intent (LOI) to OMG by September 24, 2018.
- Submit: Responses to the RFP are due by Monday, November 19, 2019.
- Vote: Organizations interested in voting on the standard, but not submitting their own proposal, must join OMG by Monday, November 19, 2019.

Next Steps

- Watch the webinar: https://www.brighttalk.com/webcast/12231/270815

Read the RFP, Download the LOI template and Learn more: http://www.omgsysml.org/SysML-2.htm

---

4.7 Project Performance International (PPI) and the International Council on Systems Engineering (INCOSE) Partner to Develop a Systems Engineering Tools Database

PPI to Share Access to its Systems Engineering Goldmine with INCOSE Members

via the INCOSE Website

MELBOURNE – September 7, 2018 – The International Council on Systems Engineering (INCOSE), the largest organization in the world dedicated to systems engineering, signed a memorandum of understanding (MOU) with Project Performance International (PPI), an Australia-headquartered transnational corporation using systems engineering to improve client performance.

The MOU between PPI and INCOSE formalizes the partnership between the two parties as they develop and operate the Systems Engineering Tools Database (SETDB), an online library of systems engineering software tools data to support engineering and business processes. INCOSE and PPI's partnership also includes the sharing of PPI's Systems Engineering Goldmine, a resource of over 4GB of systems engineering documents and a searchable database of definitions of more than 7,800 terms used in systems engineering. The collaboration also embraces the delivery of an educational seminar series for infrastructure professionals.

"This collaboration between PPI and INCOSE is another step towards PPI's vision of all engineers worldwide practicing systems engineering, as the way that engineering is done. We at PPI look forward with excitement to working more closely with INCOSE in the future, to the benefit of humankind," said PPI's Managing Director Mr. Robert Halligan.

"INCOSE and PPI have a mutual commitment to improving the state of systems engineering practice. These projects help us to achieve our shared goal and to bring the systems engineering community closer together as we continue to innovate," said Garry Roedler, President of INCOSE.

The joint development of the Systems Engineering Tools Database will be hosted on an INCOSE server linked to the INCOSE and PPI websites, and will provide current and future members of INCOSE and clients of PPI with data on systems engineering software tools.

To learn more about PPI and its Systems Engineering Goldmine visit https://www.ppi-int.com and https://segoldmine.ppi-int.com.

For more information about INCOSE or to join its growing community of systems engineers, visit www.incose.org.

About Project Performance International

Project Performance International (PPI) is a for-profit company incorporated in Australia and operating worldwide. The mission of PPI is to improve the performance of its clients and the lives of their people by improving the practice of engineering, based on systems thinking, and using the principles and methods of systems engineering. Learn more about PPI at www.ppi-int.com.

About the International Council on Systems Engineering

The International Council on Systems Engineering (INCOSE) is a not-for-profit membership organization that promotes international collaboration in systems engineering practice, education and research. INCOSE's mission is to "address complex societal and technical challenges by enabling, promoting and advancing systems engineering and systems approaches." Founded in 1990, INCOSE has more than 70 Chapters and over 16,500 members worldwide. For additional information about INCOSE, call 1-858-541-1725 or visit www.incose.org. Become a member today.

5. FEATURED ORGANIZATIONS

5.1 Centre for Science and Policy University of Cambridge

The Centre for Science and Policy (CSaP) at the University of Cambridge promotes relationships between policy professionals and experts in the sciences and engineering by:

- Providing policy professionals access to academic thinking in engineering, science, computing, mathematics, the social sciences, law and philosophy.
- providing training, support and opportunities for researchers to engage with policy makers.
- Providing an arena in which those interested in the policy implications of the sciences and technology, and the relationship between research expertise and public policy, can discuss and develop fresh ideas.

CSaP works with a unique network of academicians and decision-makers to improve the use of evidence and expertise in public policy. The approach is based on addressing the questions which policy makers identify, and on building relationships characterized by mutual understanding, respect, and trust.

More Information

5.2 International Society for System Sciences

The International Society for the Systems Sciences is a worldwide organization for systems sciences. Initially conceived in 1954 as the Society for the Advancement of General Systems Theory, and started in 1955/56, the Society for General Systems Research became the first interdisciplinary and international co-operation in the field of systems theory and systems science. In 1988 the society adopted its current name to reflect its broadening scope. Access the site through the following link: Journals ISSS for online conference proceedings, downloadable PDFs of papers, and the full conference program and abstract book. These official publications of the ISSS are submitted by authors, peer-reviewed, and published for access at no cost to viewers. The ISSS Journal carries the ISSN 1999-6918.

More Information

5.3 ARC Advisory Group

Founded in 1986, ARC Advisory Group is a leading technology research and advisory firm for industry, infrastructure, and cities. ARC provides in-depth coverage of both information technologies (IT) and operational technologies (OT) and associated business trends. Analysts and consultants provide industry knowledge and experience. ARC provides technology supplier clients with strategic market research and helps end user clients develop appropriate adoption strategies and evaluate and select the best technology solutions for their needs.

More Information

5.4 SysAdmin, Audit, Network, and Security (SANS) Institute

The SANS Institute is a private U.S. for-profit training company founded in 1989 that specializes in information security, cybersecurity training and technical certification in information security. Topics available for training include cyber and network defenses, penetration testing, incident response, digital forensics, and audit. The information security courses are developed through a consensus process involving administrators, security managers, and information security professionals. The courses cover security fundamentals and technical aspects of information security. The Institute has been recognized for its training programs and certification programs. SANS stands for SysAdmin, Audit, Network and Security.

More Information

5.5 Defense Advanced Research Projects Agency (DARPA)

The Defense Advanced Research Projects Agency is an agency of the United States Department of Defense responsible for the development of emerging technologies for use by the military. Originally known as the Advanced Research Projects Agency, the agency was created in February 1958 by

President Dwight D. Eisenhower in response to the Soviet launching of Sputnik 1 in 1957. Since its inception, the agency's mission is ensuring that the United States avoids further technological surprise. By collaborating with academic, industry, and government partners, DARPA formulates and executes research and development projects to expand the frontiers of technology and science, often beyond immediate U.S. military requirements. (Wikipedia)

More Information

---

6. NEWS ON SOFTWARE TOOLS SUPPORTING SYSTEMS ENGINEERING

6.1 AgileCraft: Scaled Agile Management Platform

by

Alwyn Smit

Principal Consultant & Course Presenter

Project Performance International (PPI)

Email: asmit@ppi-int.com

In SyEN64 we provided a short overview concerning the Scaled Agile Framework®. AgileCraft is claimed by its developers to be the only platform built from the ground-up for “scaled lean/agile excellence”. As a Scaled Agile Management (SAM) platform, the cloud architecture makes it possible to visualize, manage and execute work at four layers of scale. Private cloud and on premise options are also available.

The software also includes the ability to connect to the tools you already have in place within your team, or to add new ones, and scale them across the other layers of the framework. A full set of team tools that are provided for can be found at https://agilecraft.com/connector.

AgileCraft enables teams to make connections between their work and company strategy and the overall big picture, something that often gets lost in an agile environment and as a result, they lose sight of the massive value they provide towards the big picture.

In a recent report by Gartner on critical capabilities for enterprise agile planning tools, they had this to say about Agilecraft:

“A new vendor in this Critical Capabilities report, AgileCraft offers a comprehensive solution for large-scale organizations aspiring to embrace agile and DevOps. It has built-in support for multiple enterprise agile frameworks, including SAFe. AgileCraft can roll up data from teams using various methodologies and team-level tools. It can track software changes from ideation through the DevOps toolchain to release.”

AgileCraft ranks at or near the top for use cases that need the breadth of functionality that the product contains. It scores well for support of all three portfolio use cases. It has the top ranking for large traditional project and program portfolios, and also is the top ranked vendor for support of SAFe.

AgileCraft offers good support for single-team use cases for both Scrum and lean/Kanban and can thus be used by teams transitioning to agile and continuing on to broad enterprise adoption. It can also be used on top of other team-level solutions in a federated solution.

7. SYSTEMS ENGINEERING PUBLICATIONS

7.1 Modeling and Simulation Support for System of Systems Engineering Applications

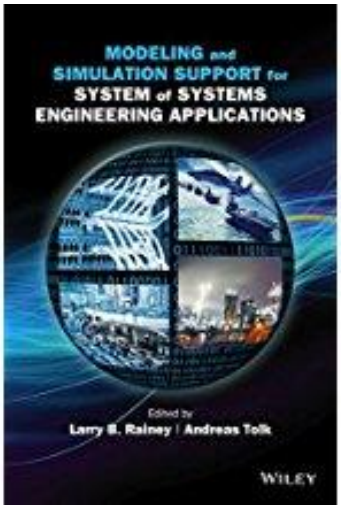

Image source

by

Larry B. Rainey and Andreas Tolk

From the Amazon.com Website:

"...a much-needed handbook with contributions from well-chosen practitioners. A primary accomplishment is to provide guidance for those involved in modeling and simulation in support of Systems of Systems development, more particularly, guidance that draws on well-conceived academic research to define concepts and terms, that identifies primary challenges for developers, and that suggests fruitful approaches grounded in theory and successful examples." Paul Davis, The RAND Corporation

Modeling and Simulation Support for System of Systems Engineering Applications provides a comprehensive overview of the underlying theory, methods, and solutions in modeling and simulation support for system of systems engineering. Highlighting plentiful multidisciplinary applications of modeling and simulation, the book uniquely addresses the criteria and challenges found within the field.

Beginning with a foundation of concepts, terms, and categories, a theoretical and generalized approach to system of systems engineering is introduced, and real-world applications utilizing case studies and examples are presented. A unified approach is maintained in an effort to understand the complexity of a single system as well as the context among other proximate systems. In addition, the book features:

- Cutting edge coverage of modeling and simulation within the field of system of systems, including transportation, system health management, space mission analysis, systems engineering methodology, and energy.
- State-of-the-art advances within multiple domains to instantiate theoretic insights, applicable methods, and lessons learned from real-world applications of modeling and simulation.
- The challenges of system of systems engineering using a systematic and holistic approach.
- Key concepts, terms, and activities to provide a comprehensive, unified, and concise representation of the field.
- A collection of chapters written by over 40 recognized international experts from academia, government, and industry.
- A research agenda derived from the contribution of experts that guides scholars and researchers towards open questions.

Modeling and Simulation Support for System of Systems Engineering Applications is an ideal reference and resource for academics and practitioners in operations research, engineering, statistics, mathematics, modeling and simulation, and computer science. The book is also an excellent course book for graduate and PhD-level courses in modeling and simulation, engineering, and computer science.

Format: Hardcover

Publisher: Wiley; 1 edition (February 9, 2015)

ISBN:

ISBN-10: 1118460316

ISBN-13: 978-1118460313

More Information

7.2 Deep Shift: Technology Tipping Points and Societal Impact

Image source

provided by

Global Agenda Council on the Future of Software & Society
World Economic Forum

Software has the potential to drastically change our lives. In 2016, the World Economic Forum's Global Agenda Council on the Future of Software and Society set out to help people prepare for changes enabled by software. The Council identified 21 examples that will have far-reaching impacts on human health, the environment, global commerce, and international relations.

We are entering a time of momentous societal shifts brought on by advancements in software. According to Erik Brynjolfsson, Council Vice-Chair; Director, MIT Initiative on the Digital Economy, Massachusetts Institute of Technology, USA, and a prolific author: "Now comes the second machine age. Computers and other digital advances are doing for mental power – the ability to use our brains to understand and shape our environments – what the steam engine and its descendants did for muscle power."

These changes will impact people around the world. Inventions such as artificial intelligence, connected devices and 3D printing will enable us to connect and invent in ways we never have before. Businesses will automate complicated tasks, reduce production costs, and reach new markets. Continued growth in internet access will further accelerate change. In Sub-Saharan Africa and other underdeveloped regions, connectivity has the potential to redefine global trade, lift people out of poverty, and topple political regimes. For many of us, seemingly simple software innovations will transform our daily routines. These changes are not without their challenges; as technology improves the lives of many, the Council hopes to help prepare people to understand and address concerns on privacy, security, and job disruption.

This report is a small first step in understanding the changes that lie ahead. It provides important insights to consider as we navigate the complex issues related to changing technologies.

Download the report

7.3 Notes on the Synthesis of Form

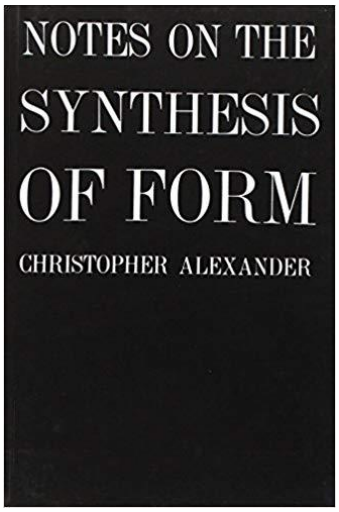

Image source

by

Christopher Alexander

"These notes are about the process of design: the process of inventing things which display new physical order, organization, and form, in response to function." This book, opening with these words, presents an entirely new theory of the process of design.

In the first part of the book, Alexander discusses the process by which a form is adapted to the context of human needs and demands that has called it into being. He shows that such an adaptive process will be successful only if it proceeds piecemeal instead of all at once. It is for this reason that forms from traditional unselfconscious cultures, molded not by designers but by the slow pattern of changes within tradition, are so beautifully organized and adapted. When the designer, in our own self-conscious culture, is called on to create a form that is adapted to its context he is unsuccessful, because the preconceived categories out of which he builds his picture of the problem do not correspond to the inherent components of the problem, and therefore lead only to the arbitrariness, willfulness, and lack of understanding which plague the design of modern buildings and modern cities.

In the second part, Alexander presents a method by which the designer may bring his full creative imagination into play, and yet avoid the traps of irrelevant preconception. He shows that, whenever a problem is stated, it is possible to ignore existing concepts and to create new concepts, out of the structure of the problem itself, which do correspond correctly to what he calls the subsystems of the adaptive process. By treating each of these subsystems as a separate sub problem, the designer can translate the new concepts into form. The form, because of the process, will be well-adapted to its context, non-arbitrary, and correct.

The mathematics underlying this method, based mainly on set theory, is fully developed in a long appendix. Another appendix demonstrates the application of the method to the design of an Indian village.

One reviewer’s comments concerning this book:

Anyone who designs things should read this book. It's architecture: the thought processes of moving through the phases of panic and ignorance to discovery and fascination to the magical moment when your design begins to talk back to you, and tell where YOU need to go, and when you've made a mistake, or when your design shows you a brilliant idea that never would have occurred to you. There are virtually no books on the joys and angst of the design thought process, so this book is priceless.

Format: Hardcover and paperback

Publisher: Harvard University Press (October 24, 1964)

ISBN:

ISBN-10: 0674627512

ISBN-13: 978-0674627512

More Information

7.4 The 'How' of Transformation

by

Michael Bucy, Adrian Finlayson, Greg Kelly, and Chris Moye

McKinsey & Company

Disruptive forces abound in today's business environment. Technological innovation, regulatory changes, pressure from activist investors, and new entrants are just some of the forces causing disruption, even in historically less volatile business sectors. It's therefore no surprise that many consumer-goods and retail companies are embarking on transformation efforts, sometimes in response to outside pressure and other times to get ahead of it. Regardless of why, these companies are introducing new ways of working to large numbers of employees, with the goal of producing a step-change, sustainable boost in business results.

However, the painful reality is that most transformations fail. Research shows that 70 percent of complex, large-scale change programs don't reach their stated goals. Common pitfalls include a lack of employee engagement, inadequate management support, poor or nonexistent cross-functional collaboration, and a lack of accountability. Furthermore, sustaining a transformation's impact typically requires a major reset in mind-sets and behaviors—something that few leaders know how to achieve.

About the Authors

Michael Bucy is a partner in McKinsey's Charlotte office; Adrian Finlayson, based in Melbourne, Australia and Chris Moye, based in Philadelphia, USA are senior vice presidents in McKinsey's RTS, a McKinsey unit focused on supporting turnarounds and transformations across industries worldwide; and Greg Kelly, a director in the Atlanta office, is the global leader of McKinsey's Consumer Packaged Goods and Retail Practices.

Access the article

7.5 If I Knew I was going to be a PM, I Would Have Followed a Different Path

by

Mark Mullaly

Mark Mullaly is president of Interthink Consulting Incorporated, an organizational development and change firm specializing in the creation of effective organizational project management solutions. Since 1990, it has worked with companies throughout North America to develop, enhance and implement effective project management tools, processes, structures, and capabilities. Mark was most recently co-lead investigator of the Value of Project Management research project sponsored by PMI. You can read more of his writing at *www.markmullaly.com*

From the article:

Many of us fall into the project management role by accident. We don't set out with the deliberate intent to have the title "project manager" printed on our business cards (or to have the role indelibly carved into our psyche). For all that project management is about being proactive, many of us wind up taking a far more reactive and unplanned path into the project management world. Then something happens that wakes us up to the idea that there are concepts, terms and approaches that can help us get stuff done better.

Because we are often accidental project managers, we don't just take an indirect route into the role. We also typically take a circuitous path to skills development as well. This is true even for people coming into the workforce today. While they may be exposed to the idea of projects, it is still rare and atypical for project management skills and techniques to be taught as part of formal education.

In terms of development, I followed two primary paths to learning about project management. First, I read a lot. And second, I learned by doing. I experimented and tried things out. Sometimes those attempts worked, and I kept at them. Often they didn't work, and I adapted or abandoned them in favor of other approaches. I made mistakes, and course-corrected as necessary. And I tried to learn from the mistakes of others.

For me, project success and managing projects well really isn't about the mechanics. What makes projects messy and challenging and difficult is people. I rely on negotiation skills, an appreciation of human nature, and what I hope to be a sufficient level of emotional intelligence and empathy. I leverage strategy, problem solving, decision making, and facilitation techniques. I talk some, and I try to listen far more. I listen for meaning, test for comprehension, and don't shy away from ambiguity or disagreement.

The development opportunity—and the challenge—is that the breadth of skills I am talking about here is enormous. They don't consciously connect with each other, although there are most assuredly overlaps.

There is no single map of what they are, how to build them, or which particular sequence might be useful. We need to become aware of a tool, skill, or approach. We need to register its interest and relevant. We need to expose ourselves to the philosophies, principles, and practices that define how they work. And we need to reflect on how they are relevant, how we might apply them (and in what circumstances), and how they connect to all of the other capabilities, skills, and perspectives that we already have.

More Information

7.6 Why Do Projects Fail?

by

Arvind Kumar

Kumar has more than 13 years of progressive experience in software development, which includes project management, business analysis and planning, development, implementation, and quality orientation.

From the Article:

In a perfect world, every project would be on time and within budget. But reality—especially when viewed through statistics—tells a very different story. It's not uncommon for projects to fail. Even if the budget and schedule are met, one must ask, "Did the project deliver the results and quality we expected?"

True project success must be evaluated on all three components. Otherwise, a project could be considered a failure. There are many reasons why projects (both simple and complex) fail.

Some of the common reasons why projects fail, based on my experience, are the following:

- Incompetent project managers.
- Improper involvement level of PMs.
- Lack of management support.
- Ignoring change management processes.
- No risk management process.
- Inaccurate stakeholder analysis.
- Missing project management tools.
- Always saying “yes” to the customer.
- Not talking through problems.
- Guesstimating.

- Avoiding code reviews.
- Skipping prototyping.

There are many seemingly independent causes of project failure. One way to overcome these causes is for the various stakeholders to be included in a very thorough planning process, thereby maximizing input from various vested interests—and broadening the understanding of the project manager and team members. An improvement in the success rate of projects is possible by putting significantly more focus on general management activities. With accurate planning, defined goals, clear assignments, and effective communication, proactive managers can overcome those odds to master even the most challenging project.

More Information

7.7 How to Save a Failing Project: Chaos to Control

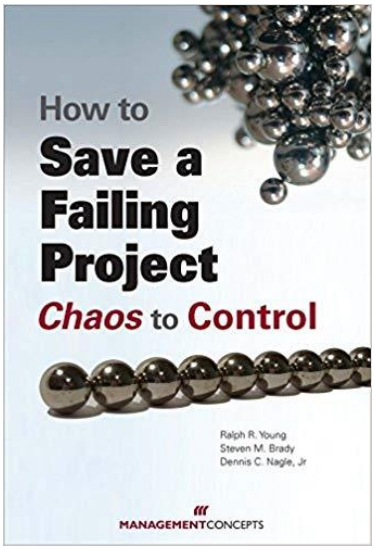

Image source

by

Ralph R. Young, Steven M. Brady, and Dennis C. Nagle, Jr.

From the Amazon.com Website:

"HOW TO SAVE A FAILING PROJECT: CHAOS TO CONTROL by Ralph R. Young et al is a competently written book with a misleading title. It should have been called HOW TO KEEP A PROJECT FROM FAILING, since it's a collection of good project management advice from a voice of experience. It's well-organized and well-presented and could probably replace dozens of its predecessors already lining the project management bookshelves, covering this territory before but not as well. If I were teaching project management, I'd certainly consider using this book in class. Planning, team-building, managing

expectations, sharing a vision--it's all there, and the authors have obviously been there and done that." Susan de la Vergne

You CAN Turn Around A Failing Project! Poor project results are all too common and result in dissatisfied customers, users, and project staff. With countless people, goals, objectives, expectations, budgets, schedules, deliverables, and deadlines to consider, it can be difficult to keep projects in focus and on track. How to Save A Failing Project: Chaos to Control arms project managers with the tools and techniques needed to address these project challenges. The authors provide guidance to develop a project plan, establish a schedule for execution, identify project tracking mechanisms, and implement turnaround methods to avoid failure and regain control. With this valuable resource, you will be able to: • Identify key factors leading to failure • Learn how to recover a failing project and minimize future risk • Better analyze your project by defining proper business objectives and goals • Gain insight on industry best practices for planning.

Format: Paperback

Publisher: Berrett-Koehler Publishers; 1 edition (May 1, 2009)

ISBN:

ISBN-10: 1567262392

ISBN-13: 978-1567262391

More Information

7.8 Partnering in Construction: A Practical Guide to Project Success

by

Frank Carr and Charles D. Markert, P.E., C.P.F.

Partnering is an informal team-building construct to launch and sustain team communication, collaboration and problem-solving, during a project, between multiple parties to achieve a mutually productive outcome. It is a pragmatic approach to operationalize otherwise common platitudes and exhortations.

There are three main elements of this book. They include the lessons learned, the legal considerations, and how to do partnering.

First, the book begins with the lessons learned by the authors. These include the answers to most every imaginable question.

Second, the legal considerations were written by the legal minds who documented and memorialized partnering into the US Army Corps of Engineers policy.

Third, the 'how to do it' was written by a practicing partnering facilitator who is also a retired civil engineer (and non-lawyer).

Partnering is still in practice almost 30 years since its inception. The Associated General Contractors of America (AGC of America) adopted this version of partnering and recommends it to their members. The chief of the Army Corps of Engineers has recently admonished his field offices to increase the amount of partnering used in projects because as partnered projects have reduced in numbers, litigation has been increasing.

Partnering was astoundingly successful in reducing litigation on projects and continues to deliver benefits well beyond reduced litigation. It has also been institutionalized by many state highway departments across the country.

The same partnering described in this book has also been used for large federal contracts in the information technology world. In fact, it is applicable wherever there is a contract between two or more parties.

Copies of the book are no longer available from the publisher, the American Bar Association. However, Amazon offers both new and used copies of the book. Available here.

More Information Concerning Partnering

8. EDUCATION AND ACADEMIA

8.1 Systems Engineering at Cranfield University, UK

The Centre for Systems Engineering, Cranfield University was established in 1999 and has been at the forefront of developing systems engineering education ever since, blending the breadth of systems thinking with the rigors of systems engineering. More recently, it has taken the lead on the academic component of a masters-level apprenticeship program in Systems Engineering as part of a UK government and Defense industry partnership. The Centre is led by Professor Emma Sparks.

The MSc in Systems Engineering for Defense was established in 1991. It is a modular program offered in both part-time and full-time mode. Students have varied backgrounds, including serving military officers, civilian Ministry of Defense employees, and Defense industry, from the UK and abroad, resulting in a unique learning environment in which multiple perspectives and knowledge can be shared.

The objectives of the program are to:

- Apply systems knowledge and systems thinking to the decision-making process in relation to systems problems in a constantly changing defense environment, analyzing the principle influence and constraints in formulating a systems engineering approach.

- Evaluate the quality of systems engineering practices applied by industry and government in the defense environment, underpinned by the application of techniques, tools and processes, proposing practical systems approaches to accommodate both industrial and governmental ideologies.
- Undertake self-directed systems engineering research using the knowledge and skills acquired during the instructional phase of the course.

The course offers choice and specialization to students having different backgrounds, interests, and requirements in order to tailor the learning pathway. Group working, real world case studies, and a virtual learning environment underpin more traditional classroom-based interaction, preparing students for professional practice. The program is CEng accredited and fulfils the requirements for registration as a chartered engineer when presented with a CEng accredited bachelor's degree.

More Information

8.2 An Engineering Student's Guide to Effective Note Taking

High school might have taught you that you are allowed to pass your subjects without ever having to take down notes, but taking effective notes in lectures and tutorials is an essential skill for university study. This is especially important in engineering, as you cannot simply look at the solutions of the problems on the board or in your textbooks and expect that you have absorbed them all. This is one of the many laborious tasks that you need to practice while in engineering: note taking. Engineering students, read carefully.

Here are tips and tricks to help you in your classes with regards to note taking:

Pay attention

Active listening and thinking are what is most important. Get into the habit of absorbing and understanding what you are about to write. Clear your mind from any distraction. Strategize where you are sitting inside the classroom and stay away from anything that distracts you.

Have the right materials

Engineering students should not be allowed to enter the class without a pen and a notebook. Make sure that you have your note-taking materials with you. The use of highlighters and sticky notes may assist you further.

Take good notes in class

You are there as an engineering student, not a field reporter. Never write down everything the lecturer is saying; instead, jot down highlights of the discussion especially those written on the board. Those topics are most likely to be important. More importantly, write down information, in particular, names, dates, formulas, theories, and other details which cannot be found on the textbook. It is also about speed. You

cannot catch up on a sequence of important topics if you write entire block of sentences. Phrase and abbreviate instead to save time and prevent hand cramps.

Take good notes in readings

Part of note taking is also identifying the most important items when you go over your textbooks. Highlight or underline only text which is critical in the material. Never fill the entire page with colored pens as this will only cause confusion in your mind upon reviewing. You could jot some notes on the margins or on a loose sheet of paper to help for easy memorization.

Organize your notes

Make sure to locate the notes you have taken down when you need them. Having the course name and date at the top of the page and stapling or binding pages together also assist you. Take notes chronologically. It is important that you write as clearly as possible.

Find a system that works for you

Perhaps you have developed your own style of note taking that works well for you. Only you can decide what strategies help you to learn all the mathematical formulas and scientific principles. Experiment and find an approach that works for you!

8.3 Applications for a Visiting Associate Professor Position in Software/Systems Engineering at Carlos III University of Madrid, Spain

Full details available here.

Summary

Location: Leganés, Madrid, Spain

Contract: Full Time

Duration: 1 year + optional 3 to 5-year extension

Applications deadline: September 30, 2018

Starting date: January 21 -25, 2019

Description

The Department of Computer Science and Engineering invites applications for a Visiting Associate Professor position. Applicants must have completed a PhD in Software Engineering, Systems Engineering, Computer Science, Information Technology
or a closely related field by the application closing date.

Candidates must show a promising record in both research and teaching areas. The target areas of interest to be covered in this position include (but are not limited to):

1. Software Engineering / Systems Engineering
2. Model-based Software Development

3. Requirements Engineering
4. Knowledge-based Systems
5. Analytical Modeling and Simulation (Modelica, FMI/FMU, etc.)
6. Descriptive Modeling and Simulation (UML/SysML, etc.)
7. Information and Knowledge Retrieval and Reuse

The candidate is expected to work with researchers from both similar and different disciplines across different universities and industrial partners around the world. The candidate shall have a strong record in relevant fields of Software and/or Systems Engineering. Furthermore, a substantial external reputation shall be demonstrated with special emphasis in academic potential through high-quality research publications and participation in competitive research projects at international level. The expected candidate must have a good qualification (C1 level or equivalent) to write and read English (if not native) in order to prepare and teach courses. Knowledge of the Spanish language is not necessary.

About the Knowledge Reuse Research Group

The Knowledge Reuse (KR) research group is a multidisciplinary research group, led by Prof. Juan Llorens (http://www.linkedin.com/in/llorensjuan), with members belonging to the Department of Computer Science and Engineering (http://www.infuc3m.es/en) and the Department of Library and Information Sciences. KR's mission is to promote software and systems reuse within organizations focusing on:

- Providing research in software and systems engineering processes and methods.
- Applying semantic-based technologies to Systems Engineering practice.
- Addressing the new-technology opportunities.
- Disseminating knowledge to the scientific and industrial communities.

KR brings together these activities to enable and support people, organizations, and systems to collaborate and interoperate in the new global context. Our research lines focus on software and systems engineering applied to reuse, quality, traceability, interoperability, and reasoning, with focus on:

- Software representation. Different means to understand, transform, retrieve, and reuse software at any level.
- Systems Engineering. Exploiting knowledge-based techniques to effectively improve the communication among traditional systems engineering practices and enable better solutions to the development of complex systems.
- Ontology engineering, mainly applied to systems engineering.

The following list summarizes some on-going projects in which the group is currently involved:

- Architecture-driven, Multi-concern and Seamless Assurance and Certification of Cyber-Physical Systems (AMASS)
- Internet of DevOps (IoD)
- Interoperability Model for Land Registers (IMOLA)

Contact person

Prof. Dra. Anabel Fraga (afraga@kr.inf.uc3m.es)

For any further information about the position or the process, please contact on the following e-mail address: jobs@kr.inf.uc3m.es

9. SOME SYSTEMS ENGINEERING-RELEVANT WEBSITES

Wikipedia List of Systems Engineering

This list of systems engineering at universities gives an overview of the different forms of systems engineering (SE) programs, faculties, and institutes at universities worldwide.

https://en.wikipedia.org/wiki/List_of_systems_engineering_universities

Certification Program History

Website of the history of the certification program by INCOSE. After three years of intensive research and development, INCOSE established its Professional Certification Program in March 2004 to provide a formal method for recognizing the education, experience, and knowledge of systems engineers through the "Certified Systems Engineering Professional" (CSEP) designation. This designation requires at least five years of systems engineering experience confirmed by systems engineering knowledgeable references. Certification is valid for three years from the date awarded, and may be renewed in three-year intervals by demonstrating various ways of continuing education and ongoing professional development.

https://www.incose.org/systems-engineering-certification/certification-program-history

Road Maps

Road Maps is a self-study guide to learning system dynamics. It is organized as a series of chapters and was developed by the System Dynamics in Education Project (SDEP) at MIT under the direction of Professor Jay Forrester. This website gives access to ten chapters of Road Maps, all available for free download. Road Maps teaches the reader how to identify different kinds of systems in existence and how to model these systems. It is a resource for beginners and advanced system dynamics modelers, and requires no previous system dynamics knowledge and only basic math skills.

http://www.clexchange.org/curriculum/roadmaps/

Top Five Systems Engineering Issues in Defense Industry

A link to a .pdf report on the problems that exist in the complex defense industry, affecting both industry players as well as government participants. In this report, a Task Group was formed, inputs were solicited in advance, and a reconciliation meeting with 18 members held, in which 100 issues were raised. The group found that the bulk of the issues fell into five categories. These categories, as well as the process of carrying out the study, are detailed in this report.

https://www.aticourses.com/sampler/TopFiveSystemsEngineeringIssues_In_DefenseIndustry.pdf

---

10. STANDARDS AND GUIDES

10.1 ISO/IEC/IEEE 21839 - Systems and software engineering — System of Systems (SoS) Considerations in Life Cycle Stages of a System

ISO/IEC/IEEE 21839 - Systems and software engineering — System of Systems (SoS) considerations in life cycle stages of a system has been released to National Bodies for approval (or otherwise) of the technical content of the draft.

10.2 Cybersecurity Requirements for Industrial Control Systems Development

by

Eric Cosman,

Contributing Consultant at ARC Advisory Group

July 12, 2018

When implementing Industrial Control Systems (ICSs) to monitor and control elements of the critical infrastructure, it has always been crucial to ensure their safe and reliable operation. Traditionally, those building and operating such systems have focused on mitigating the effects of equipment failure or unexpected process conditions. In recent years, these concerns have expanded to include the possibility of accidental or deliberate compromise of the computers or networks via a cyberattack.

Engineering disciplines have defined effective and accepted normative standards for improving both safety and security. In applying these standards, asset owners have focused primarily on protecting their existing systems by first identifying and characterizing the assets requiring protection and then assembling and implementing suitable countermeasures.

Although this approach is reasonable given the size and complexity of the installed base, it is not sufficient. To make long-term gains, it is essential to address the fundamental design of ICSs. To the extent possible,

these systems must be both safe and secure by design. Cybersecurity standards are now available to address this need.

Standards Address Mitigation and Countermeasures

Over the past several years, organizations have developed standards related to ICS cybersecurity. Some target specific sectors (e.g., NERC CIP, AAMI 8001-1), while others have a broader focus (e.g., IEC 62443, UL 2900). Other sources of useful information complement the industry standards, including special publications from NIST (e.g., NIST SP800-53 and NIST SP800-82) and guidance documents from organizations like the SANS Institute (see article in the Featured Organizations section of this issue).

Many of the available standards and guidance documents are inherently reactive in that they tend to emphasize mitigating the impact of cyber incidents that could affect existing systems. These largely reactive standards are directed primarily—if not exclusively—at system integrators and asset owners. The standards state what must be done to mitigate evolving threats and vulnerabilities. The emphasis is on what is required to configure, operate, and maintain secure systems.

Lifecycle Perspective Needed

Although necessary, risk mitigation and traditional countermeasures such as malicious software detection are not enough to make lasting improvements. A truly comprehensive response must include developing and delivering new products and systems that are secure (or securable) by design. This broadens the focus to include all stages of the solution lifecycle.

Including the development stage requires involvement and commitment by component, system, and solution suppliers. Suppliers already have programs in place to respond to vulnerabilities detected in their products. To move beyond this and realize the goal of “secure by design,” they must have access to a consistent set of requirements developed and vetted by the integration providers and asset owners.

When describing the security-related performance of products, it is more accurate to use the term “securable.” This is because even the most robust products can be compromised if not configured, implemented, and supported properly. “Resilient” is another term often used in this context, implying that a product has some built-in capabilities to resist attacks while still maintaining normal functions.

Most of the major system suppliers have accepted the challenge and are working with integrators, asset owners, and other stakeholders to define the necessary requirements.

Secure Development Lifecycle

Many suppliers have taken a formal and rigorous approach to building security into their products. This process, commonly referred to as a secure development lifecycle, addresses all aspects of design, construction, testing, and vulnerability response.

Although the primary responsibility for applying such a process rests with the product or system supplier, there are opportunities for all stakeholder groups to get involved. Asset owners, regulatory agencies and standards development organizations are expected to contribute requirements based on accepted industry practice.

Standards Define Requirements

Though many suppliers employ a secure development lifecycle in creating their products, success depends on having clear requirements. Asset owners and prospective customers might provide security-related requirements as part of their selection and procurement processes, but this is not assured. Attempts to encourage this, such as the development of suggested procurement language, have been largely unsuccessful. It is common for prospective purchasers to be more focused on functional requirements, often overlooking the security needs and constraints.

Various standards have addressed this potential gap by defining a clear set of security requirements that apply to virtually all components and products. Several sector-specific standards are available for components and devices. Examples include NERC CIP, AAMI-8001-1 and parts of the UL 2900 series.

The IEC 62443 standards define security requirements that apply across sectors. They have recently been extended to address both the processes used to develop secure products as well as the product requirements.

The IEC 62443-4-1 standard describes product development lifecycle requirements related to cybersecurity for ICS products and provides guidance on how to meet the requirements described for each element.

The IEC 62443-4-2 standard provides the technical cybersecurity requirements for the components that make up an ICS (i.e., embedded devices, network components, host components and software applications). It specifies security capabilities that enable a component to mitigate threats for a given security level without the assistance of compensating countermeasures.

Both standards have been completed and approved by the ISA99 committee. They will be available both from IEC (as IEC 62443) as well as from ISA (as ANSI/ISA-62443).

Eric Cosman, a contributing consultant for the ARC Advisory Group (see article in the Featured Organizations section of this issue), has more than 35 years of experience in the development, delivery, management, and support of operations IT solutions in the process industries. During his career, his assignments and responsibilities have included process automation systems development, communications network design, functional and technical architecture design, and technology lifecycle management. He recently retired as an operations IT consulting engineer with Dow Chemical.

11. A DEFINITION TO CLOSE ON

System of Systems (from INCOSE SoS Primer)

A System of Systems (SoS) is a collection of independent systems, integrated into a larger system, that delivers unique capabilities. The independent constituent systems collaborate to produce global behavior that they cannot produce alone. This type of collaboration is powerful, but brings major challenges for systems engineering:

• Because they are independent, constituent systems may make decisions or upgrades without considering the rest of the SoS. Sometimes this unintentionally forces others to make changes too.

• Constituent systems may withdraw (possibly without warning) from the SoS if their own goals conflict with SoS goals.

• Separate constituent systems are often drawn from different engineering disciplines, and the SoS itself is commonly large-scale and usually highly complex. It is difficult to produce accurate predictive models of all emergent behaviors, and so global SoS performance is difficult to design.

• Testing and verifying upgrades to a SoS is difficult and expensive (sometimes prohibitively) due to scale, complexity and constant evolution.

INCOSE released its Primer on SoS at the INCOSE International Symposium which was held in Washington D.C. (USA) July 7 – 12, 2018. Members of INCOSE may download a free copy here.

Emerging Standard: ISO/IEC/IEEE 21839

12. CONFERENCES AND MEETINGS

For more information on systems engineering related conferences and meetings, please proceed to our website.

13. PPI AND CTI NEWS

13.1 Vale Alain Faisandier

It is with great sadness that we report the recent death from cancer of Alain Faisandier. Alain was a French SE consultant and trainer, well-known and well-liked around European SE circles. Along with Jean-Claude Roussel of Airbus and Michel Galinier of Thales, Alain set up a power base of French SE in Toulouse, and helped kickstart the Association Francaise de l'Ingénierie de Systèmes – AFIS, the French Chapter of INCOSE. This led to the European Systems Engineering Conference (EuSEC) in 2002 and the International Symposium (IS) in 2004 both being held in Toulouse, with Alain as General Chair. On the

strength of these, Alain acted as Events Director at international level for INCOSE for some years, raising the bar ever higher for the standard of each IS, and encouraging INCOSE to diversify away from its US base. In later years, Alain stepped down so that he could concentrate on writing some good textbooks, in French but also in English – definitely worth reading.

Alain had a quirky sense of humour, and was very proudly French. He could be very scathing of other countries' cuisine, and even after one sublime meal for the planning committee for the Rome IS, the best he would offer was "*C'est pas terrible*" (it's not awful). He also insulted corporate or committee stupidity in such a subtle way that it sometimes passed over the head of the intended victim. It would only be later, over a coffee or a glass of good wine, that we would hear his characteristic chuckle. I'm very sorry that I won't hear that chuckle again.

Paul Davies

13.2 PPI Participation at EnergyTech Conference 2018

Fundamentals of Systems Engineering 1-day Tutorial

Monday, 22 October 2018

This day-long tutorial initially demonstrates the business purpose of a systems approach to the engineering of things, providing abundant evidence of its high value. Then into principles and methods. Over the course of the day, you will practice, in workshop format, analysis-based methods of requirements capture and validation, model-based design, and the conduct of trade-o studies, all on a single, technology-based system. These activities will be performed within a sound conceptual framework introduced early in the tutorial. This introduction to systems engineering is aimed at any practicing engineer or engineering manager in the energy technology arena, irrespective of your prior level of experience with systems engineering.

by Robert Halligan FIE Aust CPEng IntPE(Aus)

Presentation: A Framework of Knowledge, Skills and Attitudes Conducive to High Performance Engineering *(Refer to the published technical program for timing)*

Tuesday or Wednesday, 23 or 24 October 2018

Engineering projects are notoriously challenging, no more so than projects of the types widely represented at this conference - projects that are groundbreaking, complex and critically important. We are all trying to do these projects well, and to do them better, which drives the topic of this presentation.

Engineering is a team sport, and like football, the performance of the team is governed by the the Knowledge, the Skills and the Attitudes (KSAs) of the team members. The KSAs relate to the traditional engineering streams of technology, mathematics and physics, the systems engineering stream of knowing how to go about applying the technology, mathematics and physics in a way conducive to success, and soft skills such a communication and emotional intelligence that distinguish a team from just a group.

A variety of engineering competency frameworks exists. In this presentation, we overview existing systems engineering competency frameworks. We then present a new, recommended framework; new in terms of publication but old in that it has been the foundation of the systems engineering education of over 13,000 engineers worldwide. This recommended framework exhibits a strong alignment with, we believe, practical, in-the-trenches, e cient, value-driven, high-performance engineering.

by Robert Halligan FIE Aust CPEng IntPE(Aus) and René King

Expert SE/PM Panel on PMIAA

A Panel of experts (including Virginia Greiman, Randy Ili, Robert Halligan, and Mike Pa ord) moderated by Bill Moylan discuss the current state of PM/SE Integration and the implications of the Program Management Improvement and Accountability Act (PMIAA).

Presentation: The Integration of Project Management and Systems Engineering *(Refer to the published technical program for timing)*

Thursday 25 October, 2018, morning.

This one-hour presentation on the integration of program/project management and systems engineering focusses on the more technical aspects of achieving the desired integration, including the management

framework of Project (Work) Breakdown Structure, selecting the most appropriate style of development, integrating technical project performance data with cost and schedule data, and integrating engineering decision-making with project decision-making.

by Robert Halligan FIE Aust CPEng IntPE(Aus)

13.3 Introducing Bijan Elahi – Medical Products Risk Management Leader and Educator

We are delighted to welcome Bijan Elahi to the PPI team. Bijan Elahi has authored and will deliver a new PPI course offering to the medical sector: “Medical Products Risk Management”, over three days. The first presentation of this course will be in Eindhoven, NL, over 22-24 November, 2018. See https://www.ppi-int.com/training/mdrm3d/. We hope to see you there!

Bijan Elahi is an expert on a world scale in medical devices risk management and systems engineering. He is the winner of the Educator of the Year Award from the International System Safety Society, for outstanding performance in system safety education. Mr. Elahi's personal mission is to elevate knowledge and proficiency in medical device risk management to the highest levels via teaching, coaching, and mentoring, to the benefit of companies and society. He has 25+ years of experience in risk management, working with the largest medical device companies in the world, as well as with small start-ups. He is a lecturer at Eindhoven University of Technology (the Netherlands), where he teaches risk management. The audience for this education is doctoral students in engineering as well as physicians and professionals in the medical device sector. Additionally, Mr. Elahi teaches a graduate-level course in medical device risk management at Delft University of Technology in the Netherlands.

Mr. Elahi is a principal consultant and trainer with Project Performance International.

Mr. Elahi is author of the landmark book *"Safety Risk Management for Medical Devices"*, published by Elsevier Publishing as Academic Press (3 July 2018). This is the first publication to focus exclusively on safety risk management of medical devices

The book demystifies risk management, providing clarity of thought and confidence to the practitioners of risk management as they do their work. The book delivers not only theory, but also a great deal of practical guidance for applying the theory in daily risk management work. The reader is familiarized with the vocabulary of risk management and guided through a process to ensure compliance with the international standard ISO 14971 - a requirement for all medical devices. See https://www.amazon.co.uk/Safety-Risk-Management-Medical-Devices/dp/0128130989

Mr. Elahi is a Technical Fellow at Medtronic. In this role, he teaches and consults to all Medtronic business units worldwide.

He is a contributor to the international standard ISO 14971-*"Medical devices-Application of risk management to medical devices"*. He is a member of the Editorial Board of the Journal of System Safety, a publication of the International System Safety Society, and a frequently invited speaker and lecturer at international conferences. Earlier in his distinguished career, he was a systems engineer on the Space Shuttle at NASA (USA). Mr. Elahi holds an MS Electrical Engineering degree from the University of Washington and a BS Aerospace Engineering degree from Iowa State University, United States.

His experience in medical devices spans class III implantable medical devices, electro-mechanical and disposable devices. His most recent product was a deep brain stimulator (DBS) implant for Parkinson's disease.

Mr. Elahi served as the President of the flagship chapter on INCOSE in Seattle. His other memberships include IEEE, and Tau Beta Pi, Engineering Honor Society.

Bijan lives in the Netherlands with his wife Jamie. They have two adult children. Bijan's interests include cycling, yoga, adventure travel, behavioral economics and critical thinking.

14. PPI AND CTI EVENTS

On-site systems engineering training is being delivered worldwide throughout the year. An overview of public courses is below. For a full public training course schedule, please visit https://www.ppi-int.com/course-schedule/

Systems Engineering 5-Day Courses

Upcoming locations include:

- Eindhoven, the Netherlands
- Sydney, Australia

Requirements Analysis and Specification Writing 5-Day Courses

Upcoming locations include:

- Stellenbosch, South Africa
- London, United Kingdom

Systems Engineering Management 5-Day Courses

- Upcoming locations include:
- Ankara, Turkey
- Münich, Germany

Requirements, OCD and CONOPS in Military Capability Development 5-Day Courses

Upcoming locations include:

- Amsterdam, the Netherlands
- Washington, D.C., United States of America

Architectural Design 5-Day Course

Upcoming locations include:

- Melbourne, Australia
- London, United Kingdom

CSEP Preparation 5-Day Courses (Presented by Certification Training International, a PPI company)

Upcoming locations include:

- Laurel, MD, United States of America
- Madrid, Spain

Other training courses available **on-site** only includes:

- Project Risk and Opportunity Management 3-Day
- Managing Technical Projects 2-Day
- Integrated Product Teams 2-Day
- Software Engineering 5-Day.

15. PPI UPCOMING PARTICIPATION IN PROFESSIONAL CONFERENCES

PPI will be participating in the following upcoming events. We support the events that we are sponsoring and look forward to meeting old friends and making new ones at the events at which we will be exhibiting.

INCOSE Western States Regional Conference

(Sponsoring)

Date: 20 - 22 September 2018

Location: Ogden, Utah, USA

4th IEEE Symposium on Systems Engineering

(Sponsoring)

Date: 1 - 3 October 2018

Location: Rome, Italy

INCOSE SA 2018

(Exhibiting & Sponsoring)

Date: 3 - 5 October 2018

Location: Pretoria, South Africa

INCOSE Great Lakes Regional Conference

(Sponsoring)

Date: 17 - 20 October 2018

Location: Indianapolis, IN, USA

EnergyTech Conference 2018

(Exhibiting, Tutorial, Presentation & Panel Participation)

Date: 22-25 October 2018

Location: Cleveland, OH, USA

New Zealand Defence, Industry & National Security Forum

(Exhibiting)

Date: 31 October – 1 November 2018

Location: Palmerston North, New Zealand

The INCOSE International Symposium 2019

(Exhibiting)

Date: July 2019

Location: Orlando, USA

The INCOSE International Symposium 2020

(Exhibiting)

Date;18-23 July 2020

Cape Town, South Africa

Kind regards from the PPI SyEN team:

Robert Halligan, Editor-in-Chief, email: rhalligan@ppi-int.com

Ralph Young, Editor, email: ryoung@ppi-int.com

René King, Managing Editor, email: rking@ppi-int.com

Project Performance International

2 Parkgate Drive, Ringwood, Vic 3134 Australia Tel: +61 3 9876 7345 Fax: +61 3 9876 2664

Tel Brasil: +55 12 9 9780 3490

Tel UK: +44 20 3608 6754

Tel USA: +1 888 772 5174

Web: www.ppi-int.com

Email: contact@ppi-int.com

Tell us what you think of SyEN. Email us at syen@ppi-int.info.