24th Annual INCOSE International Symposium (IS2015)
Seattle, WA, July 10 – 16, 2015

System Life Cycle Trajectories: Tracking Innovation Paths Using System DNA

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

Abstract. In-service systems change configuration across life cycles. Systems in development change in-progress developmental configurations. Evolving product lines and competing product models change configurations, over lives of product lines. Understanding system trajectories (paths of changing configurations) is important to understanding installation history, developmental progress, and competitive evolution.

What to track? For living systems, physical form (phenotype) was the initial focus, but the study of genetic information (genotype) became vital. MBSE, Configuration Management (CM), Product Lifecycle Management (PLM), Model Management (MM) disciplines and tools partially address engineered system needs. However, cyber warfare, epidemics, and other threats raise pressure to accelerate rates and efficacy of system evolution, increasing interest in agility.

This paper (Part II of a Case for Stronger MBSE Semantics) outlines "system DNA" trajectories, follows work reported by a System Sciences Working Group project at IW2014, and prepares support for the IW2015 MBSE Workshop session on patterns in Agile Systems.

Introduction

System configuration space. System configuration space is the set of all possible configurations of a system (or system family) of interest. It is made geometric by conceptualizing it as an N-dimensional space, where the N degrees of freedom describe all the system configuration variables for the system or system family of interest. Here we mean all the degrees of freedom, including stakeholder values, technical requirements, physical implementation, interfaces, normal and failure modes, and otherwise. While almost never a linear space, this configuration space can nevertheless often support inner products, projections, distance metrics, and other ideas motivated by geometric space. Basic ideas of this system configuration space used in this paper are described in another IS2015 paper (Schindel, 2015), which points out that emphasis on systems engineering process and procedure sometimes detracts from focus on this space, even though it is implicitly the ultimate target of innovation effort. This is partly over lack of its explicit representation, to the detriment of innovation performance. Model-Based Systems Engineering (MBSE) offers the opportunity to remedy this, provided the underlying semantic models supporting MBSE are strengthened.

Each point (N-tuple) in the space represents one configuration of a system. It is understood that many of these (combinatorial) configurations will be infeasible due to constraints or sub-optimal (both expressible by system models), or otherwise of less interest than other configurations. For most practical work on real systems, this N-space is not literally drawn, but used to support lower-dimensional (coupled) subspace maps and views that are themselves cross-sections of the larger configuration space.

Trajectories in System Configuration Space

Trajectories in system configuration space describe sequences of different configurations in that space:

- Travelled conceptually by human innovators designing the system, during the innovation process (for example, see Figure 1), or . . .
- Travelled in the real world by a series of configurations of an in-service system instance, over its life cycle, or . . .
- Travelled by a series of evolutionary generations of a system family, being adapted and innovated (for example, see Figure 2), or . . .
- Travelled by an ecology of systems interacting systems, including those of competitors, suppliers, and other environmental systems.

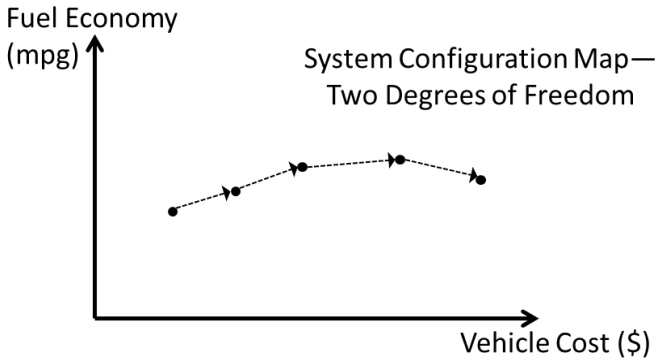


Figure 1: Path as a series of system configurations, through iterations of the SE process

Figure 2: Evolution of engineered system product lines

Sufficient Representations of System Configuration Space, and of Trajectories Within It

What are those N degrees of freedom? What are the variables? How shall we view them? Because we are interested in formal representations of systems, it should not be surprising that Model-Based Systems Engineering (MBSE) is relevant to these questions. The related question is "What is the smallest model of a system, for purposes of science or engineering?", addressed in (Schindel, 2011b). This question is equivalent to asking what the underlying

Metamodel would be for MBSE, independent of specific modeling language—it is about the underlying nature of systems. The answer offered to the question was S*Space, described by the underlying S*Metamodel, discussed in (Schindel, 2011b, 2014, 2015; Smith, Marzolf, 2014). This underlying semantic model has been applied and refined in over twenty years of MBSE applications across automotive, aerospace, telecom, medical/healthcare, consumer packaged goods, advanced manufacturing, and other domains (Berg 2014; Bradley et al, 2010; Peterson and Schindel, 2013, 2015; Cook & Schindel, 2015; Schindel & Smith, 2002).

Clearly related to explicit or implied underlying models used in various languages and interface standards, such as (OMG, 2012; U'Ren, 2003), it was reported (Schindel 2011b) that the typical resulting system representations are found to be simultaneously "too large" (redundant) and "too small" (missing key concepts). These represent opportunity for improved effectiveness as well as compression.

A "teaching summary" of some of the key concepts of the more complete S*Metamodel is shown is Figure 3, detailed further in (ICTT 2009), and also mapped to various third-party COTS engineering tools. Definitions of some of the more prominent metaclasses are listed in Appendix 1.

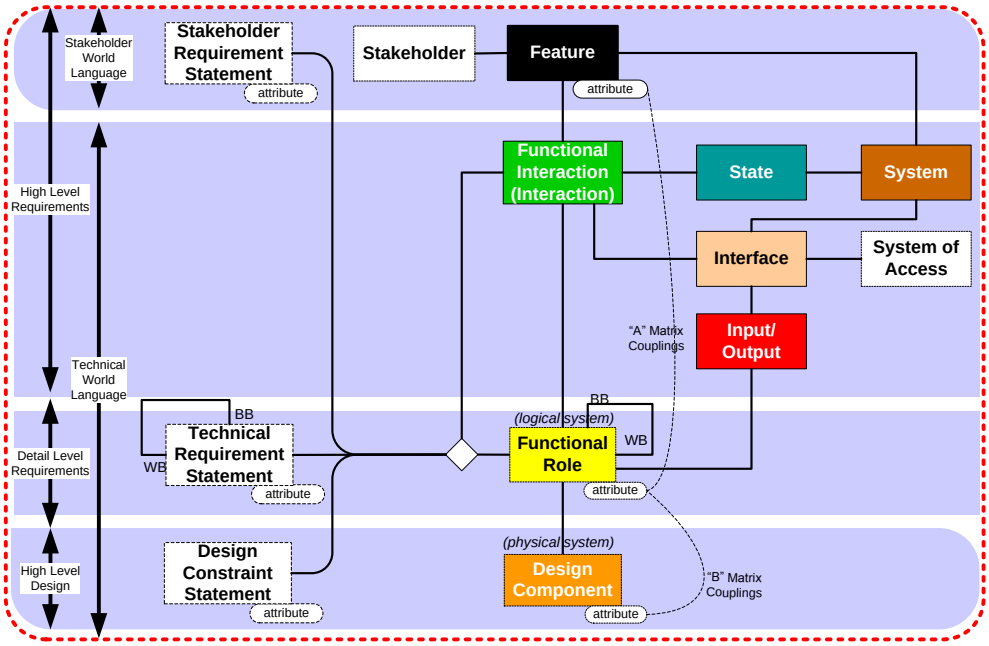


Figure 3: S*Metamodel, "Teaching Subset"

The S*Space is made up of a number of sub-spaces, including:

- Stakeholders, Stakeholder Features, and their Attributes
- Functional Interactions, Input-Outputs, and Interfaces
- Functional Roles and their Attributes
- States (Modes)
- Requirements
- Physical Components and their Attributes
- Failure Modes and Impacts
- Others

The couplings between these sub-spaces are also a key part of S*Space.

We have formally modeled the (ISO15288 2014) representation of system life cycle processes as a configurable S*Pattern, to show more explicitly how the life cycle processes (as a system) consume and produce information in the S*Space. This "System of Innovation" consumes and produces information that is in the S*Space, as symbolized by the summary diagram of Figure 4. For the system of interest being designed or supported, these life cycle processes over time result in trajectories (paths that are sequences of points) in S*Space, as in Figure 1.

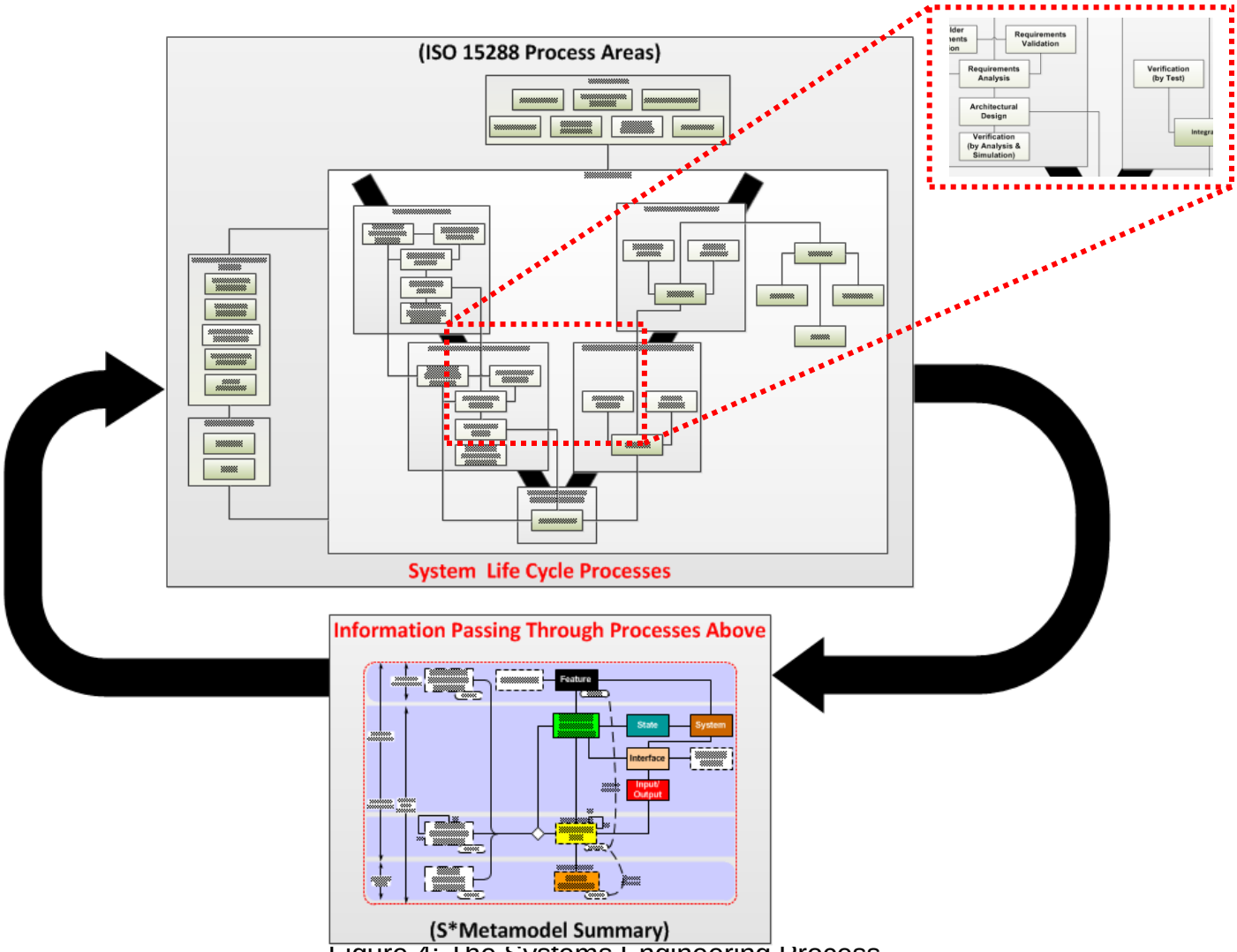


Figure 4: The Systems Engineering Process Consumes and Produces Information, All In S*Space

For examples "drilling into" the above processes in more detail, the Verification Process blocks in the above are themselves further detailed in (Cook and Schindel, 2015), where they are powered by model-based verification test patterns. Likewise, discussion of patterns optimizing human review processes in the Requirements Validation and Design Verification Processes is provided in (Nolan, Pickard, Russell, and Schindel, 2015).

Operation of the System Life Cycle Processes in the upper part of Figure 4, over time, result in movement of the system S*Configurations described by the lower part of Figure 4. The sequence of configurations in such a configuration trajectory is not typically viewed "all at once" across all the N degrees of freedom for the configured S*Metamodel (too big for human views), but that data is quite real and may be stored in one or more tool repositories.

Even though the practical views of this information are lower dimension subspace slices, having the full S*Space information formally integrated by the S*Metamodel means that we can conceptually visualize that trajectory as the sequential columns of a table of system configurations, represented by Figure 5. Each of the columns of this table represents one configuration of the "System DNA" along the path of Figure 1.

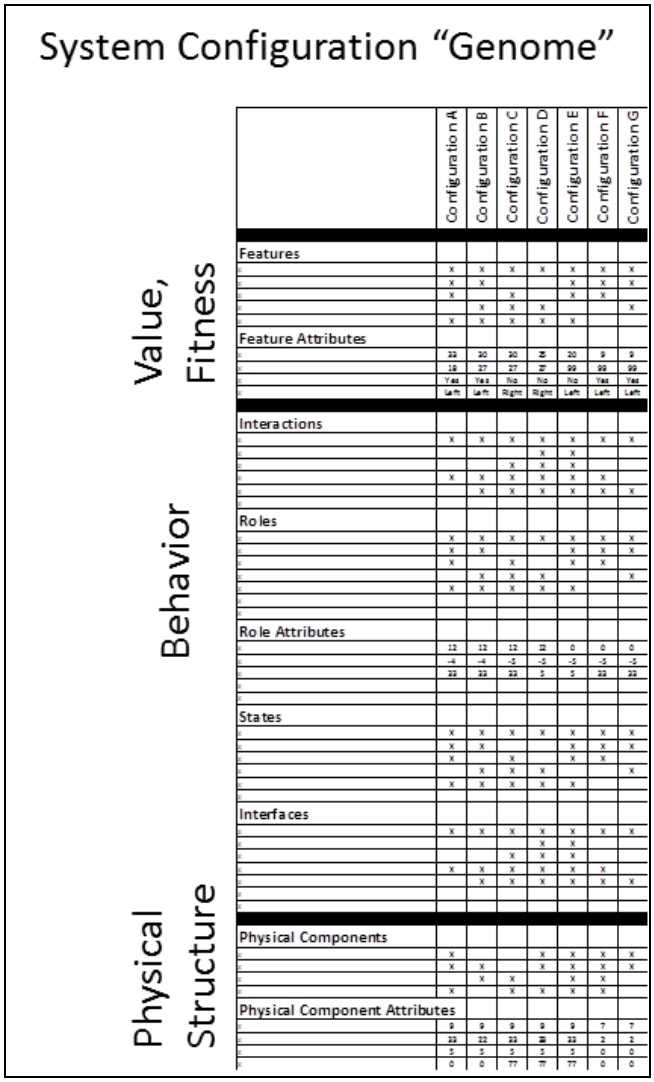


Figure 5: Progressive Configurations (Columns) Along a System Trajectory

Stakeholder Features Subspace View Has Special Significance

Figure 5 is only a conceptual perspective. For practical views of the subspaces of S*Space, each of the subspaces has its own significance--but the Stakeholder Feature subspace is first among equals. This is because it represents the value (fitness) or trade space for the rest of the S*Space—a projection of the rest of the subspaces onto a "stakeholder scoreboard", which becomes a key perspective into a project or system. For this reason, practical views in the Stakeholder Feature subspace (as in Figure 6) become important for understanding not only trade space optimization, but also failure mode effects (Schindel, 2010), risk management in general, and the basis for evaluation and decisions of all project issues.

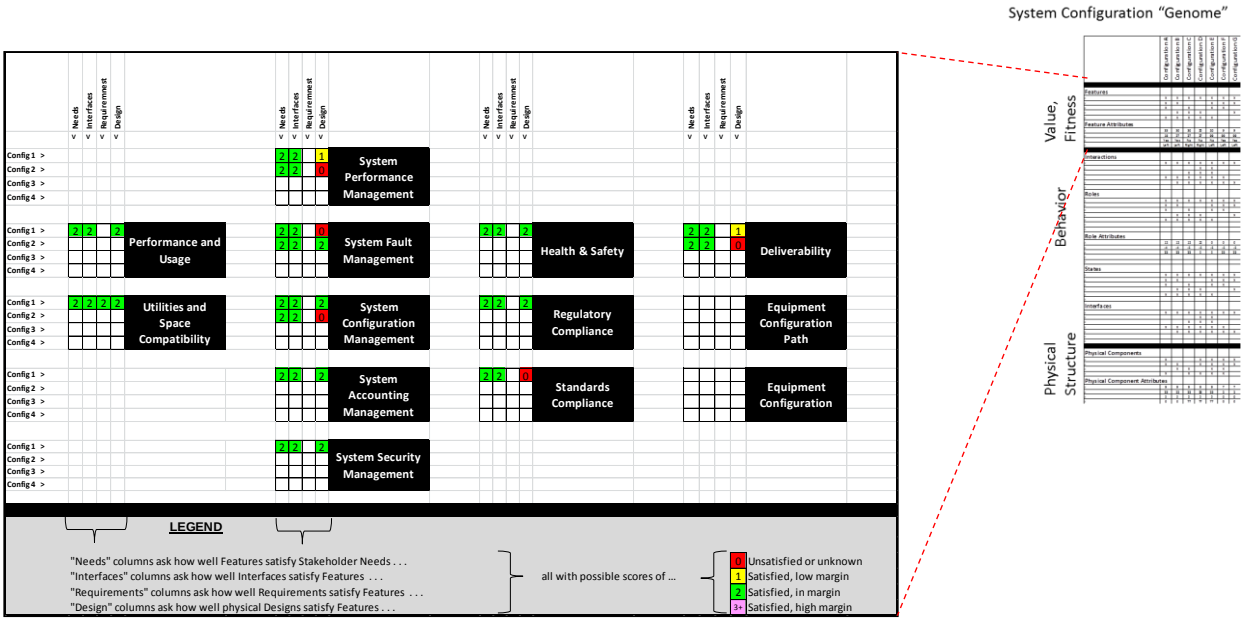


Figure 6: A View of the S*Stakeholder Feature Subspace Status

The Stakeholder Feature selection-oriented issues represent conceptual "forces" on the other sub-spaces, leading to travel through the configuration space. So, practical views of the Stakeholder Feature space take on special significance:

- Where all "whys" are represented;
- For human-engineered projects, this view is always the top level "dashboard" on progress and status;
- Highly compressible, dividing configuration vs. pattern content.

*A Differential View of Trajectories in S*Space*

Like the trajectories of other types of dynamical systems, these S*Space trajectories can be thought of in two different ways:

- Analyzed globally, as an overall path
- Analyzed locally, on a differential basis

Global views of overall paths are clearly of interest, but here are also several reasons why a differential view of system trajectories in S*Space turn out to be of special interest, discussed further below:

1. Delta Requirements
2. Compression of Path Representation
3. Equations of Motion

Delta Requirements. It is very common to see specification of requirements that are "changing" in a new system version, in comparison to past history. These might be called "Delta" or changing, requirements, and are accompanied by "Delta" of changing aspects of the other sub-spaces of the S*Space, showing changes to physical design, stakeholder features, interfaces, modes, or other aspects.

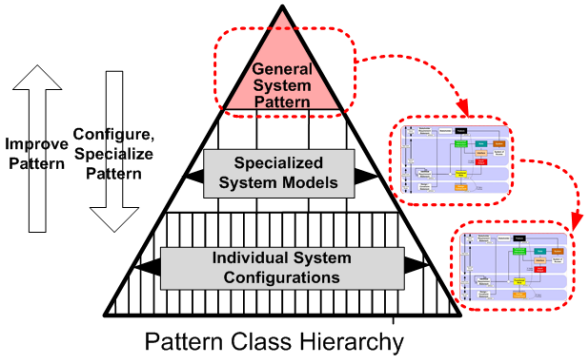
Such a "Delta" view is common in engineering because it helps call attention to what is changing and needs focal attention. But, there are (in)famous consequences of over-emphasizing these "Delta" requirements:

- Consequence 1: Some other aspect of the changing system is impacted / broken, through lack of awareness of coupled consequences.
- Consequence 2: Even if we don't break anything, by going through repeated "Delta" update cycles on a series of future versions, after many such cycles we eventually arrive at a point where no one has a description of the complete set of requirements.

What to do? When the full system "System Genome" of S*Space is tracked by information systems, it becomes more possible to "have it both ways", as follows.

Path Compression. This second subject generalizes on the special case of Delta Requirements above. Differential representations of system trajectories can further compress the dimensionality of an evolutionary path—not only for requirements, but any other aspects of the "System Genome" represented by an S*Configuration. The global path and baseline configurations can be re-created from the differential descriptions, provided they are complete—and S*Metamodel consistency helps to assure that completeness. (This is analogous to the communication engineer's Delta-Sigma Modulation.)

	Configuration A	Configuration B	Configuration C	Configuration D	Configuration E	Configuration F	Configuration G
Features							
x	x	x	x	x	x	x	x
x	x	x			x	x	x
x	x		x		x	x	
x		x	x	x			x
x	x	x	x	x	x		
Feature Attributes							
x	33	30	30	25	20	9	9
x	18	27	27	27	99	99	99
x	Yes	Yes	No	No	No	Yes	Yes
x	Left	Left	Right	Right	Left	Left	Left
Interactions							
x	x	x	x	x	x	x	x
x				x	x		
x			x	x	x		
x	x	x	x	x	x	x	
x		x	x	x	x	x	x
x							
Roles							
x	x	x	x	x	x	x	x
x	x	x			x	x	x
x	x		x		x	x	
x		x	x	x			x
x	x	x	x	x	x		
x							
x							
Role Attributes							
x	12	12	12	12	0	0	0
x	-4	-4	-5	-5	-5	-5	-5
x	33	33	33	5	5	33	33
x							
x							
States							
x	x	x	x	x	x	x	x
x	x	x			x	x	x
x	x		x		x	x	
x		x	x	x			x
x	x	x	x	x	x		
x							
Interfaces							
x	x	x	x	x	x	x	x
x				x	x		
x			x	x	x		
x	x	x	x	x	x	x	
x		x	x	x	x	x	x
x							
x							



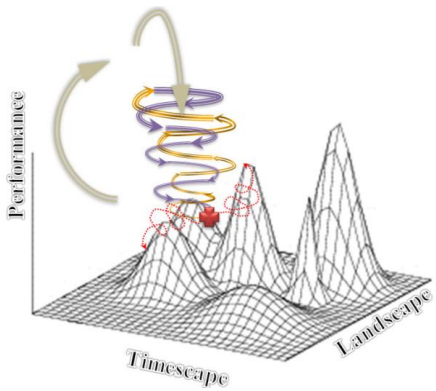


Figure 7: "Delta" Descriptions Further Compress Trajectory Representations

Equations of Motion in S*Space. Once the MBSE representation of both the System of Interest (the engineered, innovated, or supported system) and the System Life Cycle Process (the ISO 15288-like System of Innovation that acts on the System of Interest) are complete enough, a deeper understanding of the dynamical configuration trajectory becomes possible.

The S*Metamodel helps to assure that degree of completeness, adding to the case for stronger underlying semantics of systems. This subject was introduced in the IW2014 report of the System Science Working Group modeling sub-team project (Smith, Marzolf, Schindel, 2014), and will be extended in the Patterns in Agile Systems session of the MBSE Workshop at IW2015 (Dove, Schindel, 2015). In the current paper, we provide some motivating discussion of why this subject is worthy of greater attention.

Why Trajectories Are Becoming More Important: Agility in Innovation

Threats, Opportunities, Short and Long Term Trajectories. Along evolutionary paths in S*Space, versions of systems have characteristics that are different (for better or worse) than their "ancestors" (predecessors). Those ancestors may be earlier product models or biological species, but may also be earlier configurations of a current (reconfigurable) system instance, or earlier ideas in a sequence of design concepts for a single project system. In the short term, many different factors can drive movement in an S*Trajectory. In the longer term, the S*Stakeholder subspace progress determines the sustainable long-term path in S*Space.

Over multiple life cycles, systems evolve (or are evolved) in response to their environment. This includes responses to new threats and new opportunities. Figure 8 illustrates such movement in the case of the public health care system.

SATURDAY/SUNDAY, OCTOBER 25 - 26, 2014 WSJ.com

Newest Case Shifts Nation's Ebola Plan

Approach to Treating New York Doctor Influenced by Lessons From Dallas

BY BETSY MCKAY, ANA CAMPOY AND JENNIFER MALONEY

When the Centers for Disease Control and Prevention learned that a doctor in New York City likely had Ebola on Thursday, it didn't wait.

Its infectious diseases chief and a top Ebola expert were deployed hours before test results came back for Craig Spencer, who had recently returned from treating Ebola patients in West Africa. A CDC team already in the city reviewing hospitals' readiness for Ebola switched gears and pitched in to help.

The rapid response to America's latest Ebola case presents a sharp contrast to the response to the country's first case in Dallas late last month. Hospital and public-health officials are applying a host of lessons learned there to prevent any spread of Ebola in the nation's most populous city. They are aided by a public far more aware of the virus's threat than just weeks ago.

On Friday, federal and state officials sought to assure Americans that the nation remains safe. The White House said it was reviewing U.S. protocols for monitoring and quarantining travelers headed to the U.S. from the three most Ebola-stricken

Please turn to page A6

◆ Questions, praise for doctor.. A6
◆ Speeding up vaccine trials..... A6

Figure 8: Short-Term Responses to Threats and Opportunities

Co-Evolution. One may initially think of a System of Interest changing its configuration from time to time in response to relatively slow shifts in the environment of the System of Interest.

However, if that domain includes competitive systems, then it is possible that the environment may be driven to evolve faster than the System of Interest, as a competitive strategy.
In order to get this firmly in mind, it is helpful to see the entire domain system (environment) as an ecology of co-evolving systems, some of which are symbiotic (cooperative) and some of which are competitive, but all of which may be interacting with each other and moving through configuration space. For example, Figure 9 illustrates:

1. Co-evolution of hummingbirds and flowers
2. Co-evolution of radar and stealth technology
3. Co-evolution of cell phones and automobiles
4. Co-evolution of neighbouring food court stores

It is ultimately helpful to this analysis if we think of the parent system, made up of such subsystems, as itself evolving, with its own overall trajectory.

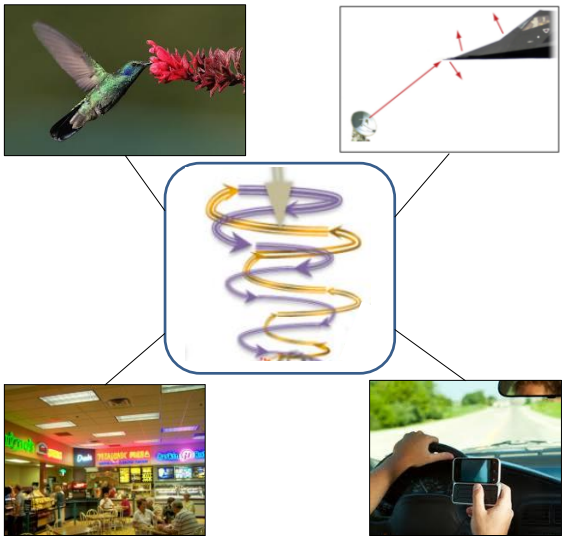


Figure 9: Co-Evolution of Interacting Systems

Is a system of interest evolving rapidly and effectively enough in response to evolution of its:

- Competitors?
- Customers?
- Prey?
- Predators?
- Opportunities?
- Threats?
- Resources?

One definition of Agile System is a system that has such a capability. Current example domains of particular concern to this line of thought include (1) Cybersecurity and the Internet of Things and (2) Epidemiological Systems.

Time of System Configuration. As human-engineered systems become more mature, their ability to be re-configured advances to later in their life cycles (see Figure 10):

1. At first, all configuration occurs during design (traditional view of design)

2. More advanced systems can be configured to order, at Manufacturing time, individually as they move through the process (Michael Dell pioneered; see also the Ford Rouge pickup truck plant)
3. Still more advanced systems can be configured after delivery, by their distributors, dealers, users, or maintainers. (e.g., manipulating configuration switches, loading data parameters)
4. Even more advanced systems can reconfigure themselves while in operation (e.g., F111 aircraft wing configuration)

Biological scientists have referred to the "evolution of evolvability" as a major step in the early stages of living systems.

Figure 10: Four Different Configuration Times During System Life Cycles

Architectural Aspects. System architectural aspects that increase system agility include:

1. Composable architecture: flexibility through configurable architecture of standard components (this and other architectural aspects are discussed in (Dove, 2014) and will be discussed at greater length in the INCOSE IW2015 MBSE Workshop joint session on Agile Systems (Dove, Schindel, 2015)). Refer to Figure 11.
2. Ability to accumulate experience as information: Cyber-Physical systems are hardware-software combinations that include information as a part of the systm architecture, discussed in (Beihoff and Schindel, 2012; Schindel, 2013b; Smith, Marzolf, Schindel, 2014) and in the following section of this paper. Refer to Figure 12.

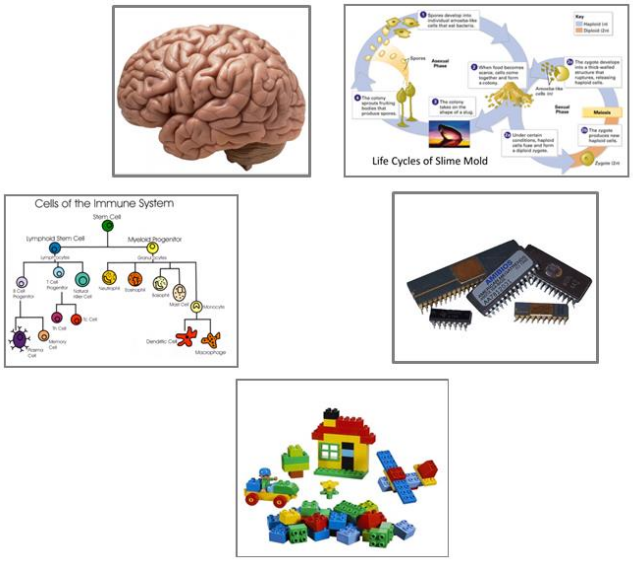


Figure 11: Composable Systems and Component Libraries

The System of Innovation Pattern, an S*Pattern discussed in the above references, describes the domain of innovation for both human-performed and other innovation processes. Figure 12 summarizes the Logical Architecture view reported for that pattern, where the Experience Accumulation Role is noted as a key behavior. Trajectories informed by the past are not the same as those which must keep re-experiencing the same "mistakes" (if they survive at all).

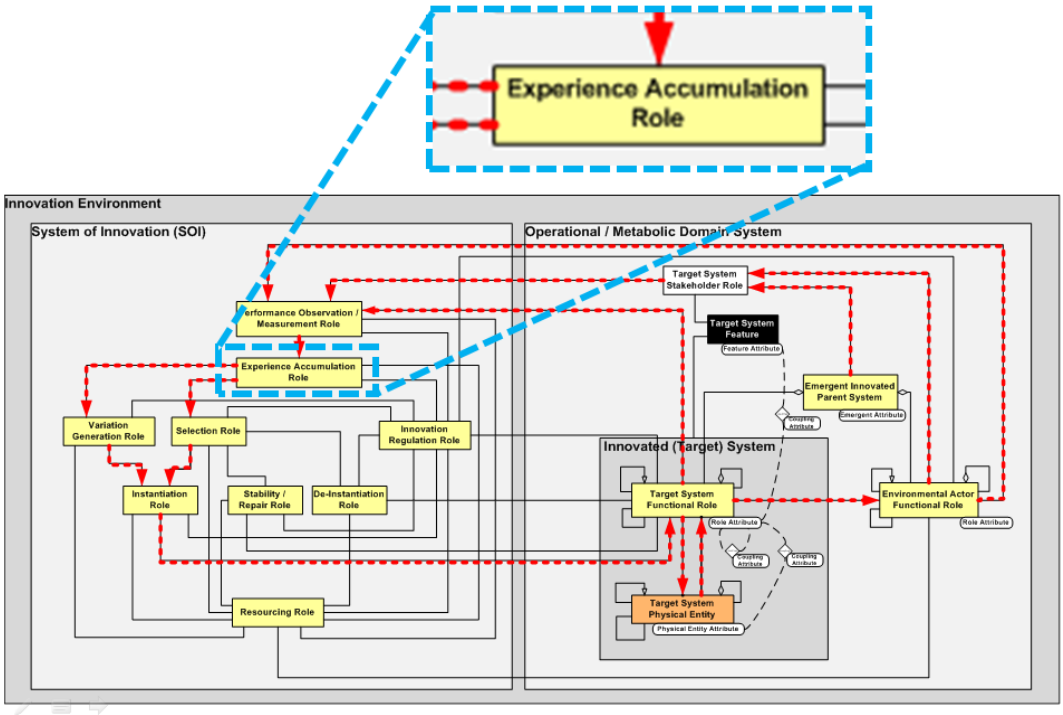


Figure 12: System of Innovation Accumulates Experience

Accumulation of Experience: Patterns as System DNA

We assert that agile (faster adapting) systems take advantage of past experience:

- An agile, composable system increases its agility if it "remembers what worked and did not" in various situations that arose in the past and might arise in the future.
- This implies learning from experience and retaining (remembering) those lessons, in some way (which could include human aspects of learning, but also other forms).

Living systems invoke previously learned modes of behavior:

- Immune systems retain memory of past antigen encounters and antibodies that worked.
- Biological DNA retains memory of protein synthesis modes that apply under various stresses.
- Brains retain memory of past situations and responses.

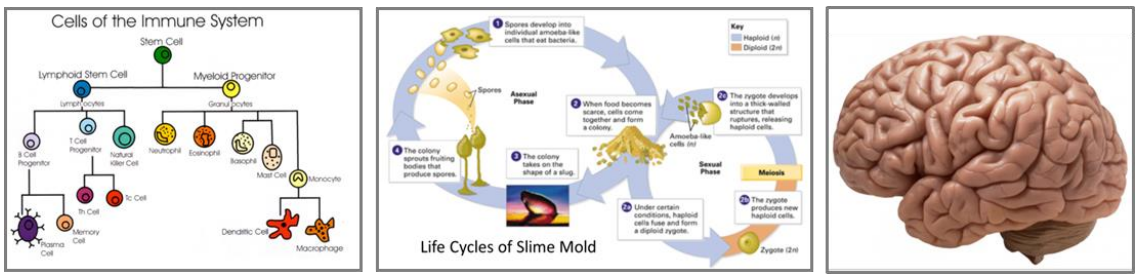


Figure 13: System Agility Enhanced by Accumulated Experience

Likewise, human designers apply their accumulated human experience to future designs:

- Personal experience, held as individuals
- Informal writings, files, libraries, attempts at formal knowledge sharing
- Pattern-based methods allow organizations to formally accumulate and reuse IP

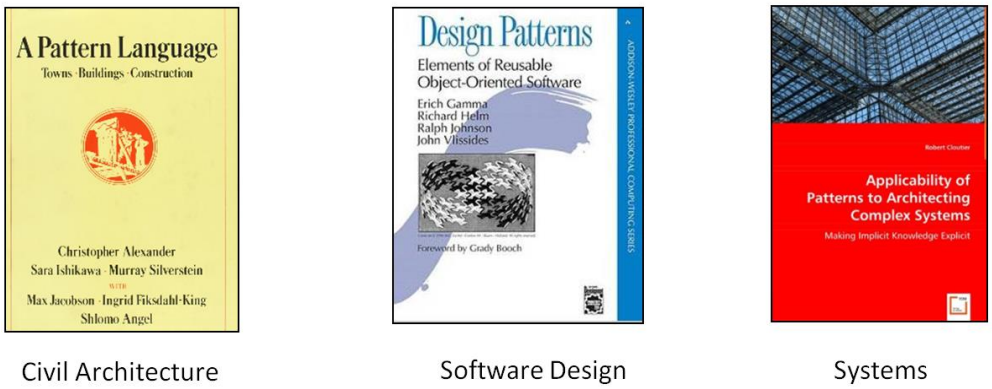


Figure 14: Formal Patterns In Human-Performed Engineering Processes

Just as biologists speak of the evolution of evolvability, there is a notion of the pattern of generating patterns. Patterns emerge from (are generated by) systems that are themselves described by the System of Innovation Pattern. Human-performed science and engineering illustrate this, but so does the natural world of innovation not performed by humans.

The discovery, formal representation, and analysis of patterns, eventually (but not at first) in the form of models, is at the heart of the physical sciences. In engineering, patterns have taken a number of forms (not all model-based in earlier patterns), in civil architecture (Alexander, 1977), software design (Gamma et al, 1995), and systems (Cloutier, 2008; Schindel, 2005a, 2007, 2011c, 2012a, 2014).

In all these cases, the expressed patterns describe regularities, across multiple instances:

- Predicting the future from the past—at least within some domain and envelope known
- Configuration space trajectories accumulate experience as patterns
- Increases the ability of (agile) systems to handle different situations
- Just having an architecture of composable components is necessary but not sufficient for high agility--we also need to know good ways to put them together, or to find those configurations soon enough when we don't know them in advance.
- Numerous examples in configurable product platforms and multi-mode systems

Agile systems are more adaptable to different situations, but "mission envelopes" apply:

- System "mission envelope" describes how widely a pattern applies.
- Adaptability, but may not anticipate refrigerators providing phone service!

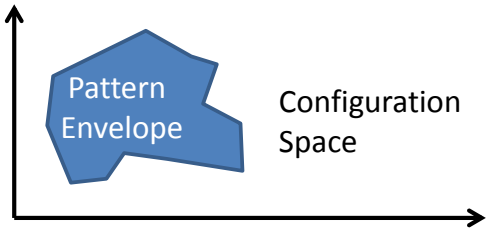


Figure 15: System Patterns Apply Across Some Defined Envelope in S*Space

The INCOSE/OMG MBSE Patterns Challenge Team (INCOSE Patterns Team 2014) is practicing the use of S*Patterns (Figure 16) as demonstrations of the "smallest possible configurable model" of adaptable systems, reported in multiple IS2015 papers (Nolan et al, 2015; Cook et al, 2015; Peterson et al; 2015; Schindel, Lewis, Sherey, Sanyal, 2015).

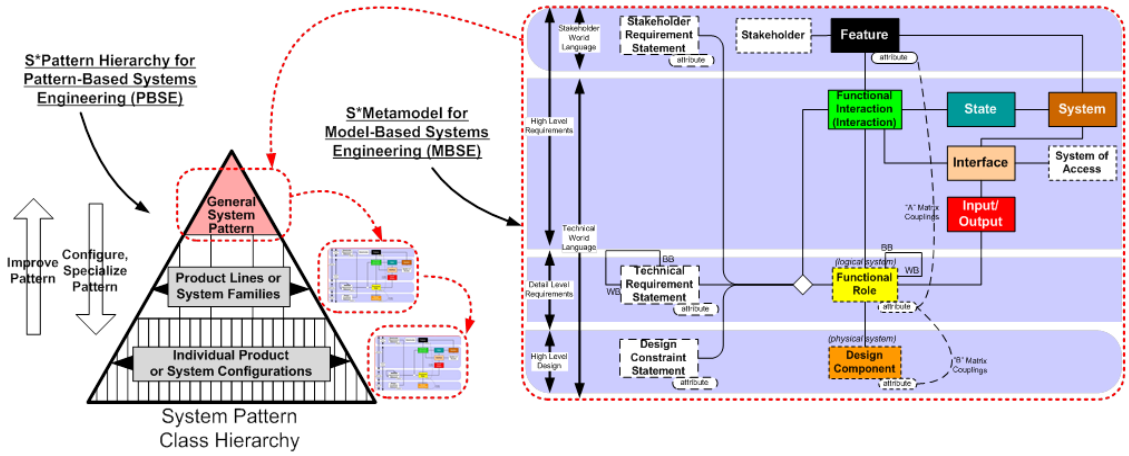


Figure 16: S*Patterns Are In Same Space as S*Models

These offer additional argument and evidence for strengthening the underlying semantic model supporting MBSE, to make it sufficient to represent S*Patterns (Schindel, 2015) and their evolving trajectories over time (this paper).

Examples of S*Patterns being worked on by Patterns Challenge Team members:

1. Integrated Enterprise Systems, Manufactured Platform Product, Manufacturing Systems, Systems of Innovation thereof
2. Systems of Verification for Safety Critical Systems
3. Automated Ground Vehicles
4. Aerospace Systems

Examples of S*Patterns in private industry:

5. Automotive and Off-Road Vehicles
6. Engines
7. Enterprise and Embedded Control Systems
8. Medical Device Products
9. Advanced Manufacturing Processes, Equipment, and Systems (Pharmaceuticals, Medical Devices, Biotech Products, Formulated Products, Aerospace Parts and Systems, Advanced Inspection Systems)
10. Advanced Packaging Systems and Packages
11. Others

Conclusions, implications and future work

1. There are very practical reasons to want to track the trajectory of system configurations, during development, during in-service life cycles, and across product line evolutions.
2. There is a minimal "genome" (S*Metamodel) that can provide a practical way to capture, record, and understand those trajectories, with significant business impact.
3. There are productive "views" of those trajectories, which may be implemented on most any general systems modeling tool or PLM system—a risk management application of

SE tracing—projecting detected gaps onto Stakeholder Feature space to understand their significance.

4. Patterns (configurable reusable models) can provide higher leverage means for implementing MBSE, tracking and exploiting system configuration trajectories, configured by selectable Stakeholder Features.
5. Joint work is underway by the INCOSE Patterns Challenge Team and Agile Systems Working Group to describe the S*Pattern representation of general Agile Systems

Appendix 1: Selected S*Metamodel Definitions

S* Metamodel Definitions. Table 2 briefly defines the metaclasses shown in Figure 2. For more information on this subject refer to (Schindel 2005a, Schindel 2011).

Table 1: Definitions of Selected S*Metaclasses (of Figure 3)

System	A collection of interacting Components. Components can be Systems.
(Functional) Interaction	An Interaction occurs when Components change each other's States by exchange of Input-Outputs.
Input-Output (IO)	Input-Outputs are energy, force, or mass (or information encoded on them), exchanged between Components during Interactions.
State	States are conditions of Components that determine their behavior in future Interactions.
Interface	Interfaces are associations of Input-Outputs, Systems of Access, and Interactions, associated with Systems, through which the Input-Outputs are said to flow.
System of Access (SOA)	Systems of Access are systems that mediate the Interaction of systems.
(Functional) Role	Roles are the behaviors performed by interacting systems.
Physical Component	Entities defined by their identity, not behavior, which may be assigned Functional Roles.
Stakeholder	People, organizations, or other entities with a stake in the performance of a System.
Feature	System behaviors, named and defined in the conceptual framework and language of Stakeholders, of value to Stakeholders.
Requirement Statement	Requirements Statements (associated with Interaction-Role pairs) describe the behavior of Roles during Interactions, in the form of (parameterized) input-output relationships.

References

1. (Alexander, 1977) Alexander, C., Ishikawa, S., Silverstein, M., Jacobson, M., Fiksdahl-King, I., and Angel, S., *A Pattern Language*. Oxford University Press, New York, 1977.
2. (Berg, 2014) Berg, E., "Affordable Systems Engineering: An Application of Model-Based System Patterns To Consumer Packaged Goods Products, Manufacturing, and Distribution", at INCOSE IW2014 MBSE Workshop, 2014.
3. (Beihoff, Schindel, 2012) Beihoff, B., and Schindel W., "Systems of Innovation I: Models of Their Health and Pathologies", Proc. of INCOSE International Symposium, 2012.
4. (Bradley, Hughes, Schindel, 2010) Bradley, J., Hughes, M. and Schindel, W., "Optimizing Delivery of Global Pharmaceutical Packaging Solutions, Using Systems Engineering Patterns" Proceedings of the INCOSE 2010 International Symposium (2010).
5. (Cloutier, 2008) Cloutier, R., Applicability of Patterns to Architecting Complex Systems: Making Implicit Knowledge Explicit. VDM Verlag Dr. Müller. 2008.

6. (Cook, Schindel, 2015) Cook, D., and Schindel, W., "Utilizing MBSE Patterns to Accelerate System Verification", to appear in *Proc. of the INCOSE 2015 International Symposium*, Seattle, WA, July, 2015.
7. (Dove, LaBarge, 2014) Dove, R., LaBarge, R., "Fundamentals of Agile Systems Engineering—Part 1" and "Part 2", INCOSE IS2014, July, 2014.
8. (Dove, Schindel, 2015) Dove, R., and Schindel, W., "Agile Modeling and Modeling Agile Systems", to appear at INCOSE IW2015 MBSE Workshop, Torrance, CA, January 24, 2015.
9. (Gamma et al, 1995) Gamma, E., Helm, R., Johnson, R., and Vlissides, J., *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley Publishing Company, Reading, MA, 1995.
10. (Gould, 2002), Gould, S., *The Structure of Evolutionary Theory*, Harvard, 2002
11. (ICTT, 2009) "Systematica Metamodel", Version 7.1, Methodology Release 4.0, May 29, 2009.
12. (ICTT, 2013) "Abbreviated Systematica 4.0 Glossary", P3125 Ver. 4.2.2, ICTT System Sciences, 2013.
13. (INCOSE SSWG 2014), INCOSE System Sciences Working Group, Systems of Innovation Project web site: https://sites.google.com/site/syssciwg/projects/o-systems-of-innovation
14. (INCOSE Patterns Team, 2014) INCOSE/OMG MBSE Initiative: Patterns Challenge Team 2013-14 Web Site: http://www.omgwiki.org/MBSE/doku.php?id=mbse:patterns:patterns
15. (ISO 15288, 2014) ISO/IEC 15288: Systems Engineering—System Life Cycle Processes. International Standards Organization (2014).
16. (Nolan, Pickard, Russell, Schindel, 2015) Nolan, A., Pickard, A., Russell, J., Schindel, W., "When two is good company, but more is not a crowd", to appear in *Proc. of the INCOSE 2015 International Symposium*, Seattle, WA, July, 2015.
17. (OMG, 2012) "OMG Systems Modeling Language, Version 1.3", Object Management Group, June, 2012.
18. (Peterson, Schindel, 2015) Peterson, T., Schindel, W., "Model-Based System Patterns for Automated Ground Vehicles", to appear in *Proc. of the INCOSE 2015 International Symposium*, Seattle, WA, July, 2015.
19. (Schindel, 2005a) Schindel, W., "Pattern-Based Systems Engineering: An Extension of Model-Based SE", INCOSE IS2005 Tutorial TIES 4, (2005).
20. (Schindel, 2007), "Are Patterns Software?", ICTT System Sciences, January 2007.
21. (Schindel, 2010) Schindel, W., "Failure Analysis: Insights from Model-Based Systems Engineering", INCOSE International Symposium, Chicago, 2010.
22. (Schindel, 2011a) Schindel, W. "Innovation as Emergence: Hybrid Agent Enablers for Evolutionary Competence" in *Complex Adaptive Systems, Volume 1*, Cihan H. Dagli, Editor in Chief, Elsevier, 2011
23. (Schindel, 2011b) Schindel, W. "What Is the Smallest Model of a System?", Proc. of the INCOSE 2011 International Symposium, International Council on Systems Engineering (2011).
24. (Schindel, 2011c) Schindel, W., "The Impact of 'Dark Patterns' On Uncertainty: Enhancing Adaptability In The Systems World", in Proc. of INCOSE Great Lakes 2011 Regional Conference on Systems Engineering, Dearborn, MI, 2011
25. (Schindel, 2012a) Schindel, W. "Introduction to Pattern-Based Systems Engineering (PBSE)", INCOSE Finger Lakes Chapter Webinar, April 26, 2012.
26. (Schindel, 2012 b) Schindel, W., "Integrating Materials, Process, & Product Portfolios: Lessons from Pattern-Based Systems Engineering", in *Proc. of Society for Advancement of Materials and Process Engineering* (SAMPE), 2012
27. (Schindel, 2013b) Schindel, W., "Systems of Innovation II: The Emergence of Purpose", *Proceedings of INCOSE 2013 International Symposium* (2013).
28. (Schindel, 2014) Schindel, W. "The Difference Between Whole-System Patterns and Component Patterns: Managing Platforms and Domain Systems Using PBSE", INCOSE Great Lakes Regional Conference on Systems Engineering, Schaumburg, IL, October, 2014
29. (Schindel, 2015) Schindel, W., "Maps or Itineraries? A Systems Engineering Insight from Ancient Navigators", to appear in *Proc. of the INCOSE 2015 International Symposium*, Seattle, WA, July, 2015.

30. (Schindel, Lewis, Sherey, Sanyal, 2015) Schindel, W., Lewis, S., Sherey, J., Sanyal, S., "Accelerating MBSE Impacts Across the Enterprise: Model-Based S*Patterns", to appear in Proc. of INCOSE 2015 International Symposium, July, 2015.
31. (Schindel, Peterson, 2013) Schindel, W., and Peterson, T. "Introduction to Pattern-Based Systems Engineering (PBSE): Leveraging MBSE Techniques", in Proc. of INCOSE 2013 International Symposium, Tutorial, June, 2013.
32. (Schindel, Smith, 2002) Schindel, W., and Smith, V., "Results of applying a families-of-systems approach to systems engineering of product line families", SAE International, Technical Report 2002-01-3086 (2002).
33. (Smith, Marzolf, Schindel, 2014), Smith G., Marzolf, T., Schindel, B., "Report of the SSWG SP/SP Modeling Sub-Team", INCOSE IW2014, Los Angeles, CA, Jan. 27, 2014.
34. (U'Ren, 2003) U'Ren, J., "An Overview of AP233: STEP's Systems Engineering Standard", ISO 10303 AP233 Working Group, October 20, 2003.
35. (Vance, 2014) Vance, A., "Updates Available", article in "Technology" section of *Bloomberg Business Week*, Sept 8-14, 2014, pp. 30-32.

Biography

Bill Schindel is president of ICTT System Sciences (www.ictt.com), a systems engineering company. His 40-year engineering career began in mil/aero systems with IBM Federal Systems, Owego, NY, included service as a faculty member of Rose-Hulman Institute of Technology, and founding of three commercial systems-based enterprises. He has led and consulted on improvement of engineering processes within automotive, medical/health care, manufacturing, telecommunications, aerospace, and consumer products businesses. Schindel earned the BS and MS in Mathematics. He co-led a 2013 research project on the science of Systems of Innovation within the INCOSE System Science Working Group, and currently co-leads the Patterns Challenge Team of the OMG/INCOSE MBSE Initiative. Bill is an INCOSE CSEP, and president of the Crossroads of America INCOSE chapter.