2014 NDIA GROUND VEHICLE SYSTEMS ENGINEERING AND TECHNOLOGY SYMPOSIUM
SYSTEMS ENGINEERING (SE) TECHNICAL SESSION
AUGUST 12-14, 2014 - NOVI, MICHIGAN

Pattern Based Systems Engineering – Leveraging Model Based Systems Engineering for Cyber-Physical Systems

Bill Schindel
President
ICTT System Sciences
Terre Haute, IN 47803

Troy Peterson
Fellow & Chief Engineer
Booz Allen Hamilton
Troy, MI 48084

ABSTRACT

As a network of interacting elements, cyber-physical systems (CPS) provide tremendous opportunities to advance system adaptability, flexibility and autonomy. However, they also present extremely complex and unique safety, security and reliability risks. The Department of Defense is seeking methods to deliver and support trusted systems and manage risks associated with mission-critical functionality. Technical thought leaders have discussed the need to address 10:1 more complex systems with 10:1 reduction in effort, using people from a 10:1 larger community than the "systems expert" group. This paper briefly summarizes the approach of Pattern-Based Systems Engineering (PBSE), which leverages the power of Model-Based Systems Engineering (MBSE) to rapidly deliver these benefits to the larger systems community. This order-of-magnitude improvement is especially necessary to address the rapidly increasing complexity of today's and future cyber-physical systems. While applying PBSE expresses many patterns, this paper introduces the Embedded Intelligence (EI) Pattern, particularly relevant to cyber-physical systems such as autonomous ground vehicles.

INTRODUCTION

The National Science Foundation (NSF) defines cyber-physical systems (CPS) as "engineered systems that are built from, and depend upon, the seamless integration of computational algorithms and physical components"[1]. They are systems which tightly intertwine computational elements with physical entities within aerospace, automotive, energy, healthcare, manufacturing and other sectors.

The rapid evolution of CPS across sectors is driving competition and innovation in a reinforcing loop. The Boston Consulting Group's 2013 report on the most innovative companies noted that for the first time, there were more automakers than tech companies listed the top 20[2]. Given that Cyber Physical systems permeate our society today, it is no surprise that the majority of the companies listed are intimately involved with Cyber Physical Systems as a core element of their business. One surprise, however, was that every company listed in the top 10 is involved with autonomous vehicles, an especially complex type of CPS of particular interest to the DoD ground community.

The rapid increase in complexity within the ground systems community is changing the way we develop, manage and interact with systems. CPS places significant demands on organizations to ensure rigor and trustworthiness of systems by improving safety, security and reliability. The NSF notes that CPS challenges and opportunities are both significant and far-reaching. To address these challenges the NSF is calling for methods to conceptualize and design for the deep interdependencies inherent in CPS.

While Cyber Physical Systems advance and transform the landscape, the Systems Engineering discipline is also experiencing a transformation—moving from a document-based to model-based approach. This transition is necessary to advance the discipline and handle the complexity and emergent behaviors exhibited by CPS. This paper will introduce Pattern-Based Systems Engineering (PBSE), the emergence of the Embedded Intelligence (EI) Pattern for Cyber-Physical Systems, and highlight its key implications and benefits as applied to the development of CPS.

MODEL-BASED SYSTEMS ENGINEERING (MBSE)

The systemic complexity of CPS demands a systems engineering approach. It requires a systems paradigm which is interdisciplinary, leverages principals common to all complex systems and applies the requisite physics-based and mathematical models to represent them. INCOSE defines Model-Based Systems Engineering (MBSE) as "the formalized application of modeling to support system requirements, design, analysis, verification and validation activities beginning in the conceptual design phase and continuing throughout development and later life cycle phases…[3]" The Object Management Group's MBSE wiki notes that "Modeling has always been an important part of systems engineering to support functional, performance, and other types of engineering analysis.[4]"

The application of MBSE has increased dramatically in recent years and is becoming a standard practice. This has been enabled by the continued maturity of modeling languages such as SysML and significant advancements made by tools vendors. These advancements are improving communications and providing a foundation to integrate diverse models. However, MBSE is often discussed as being composed of three fundamental elements – tool, language and method. This third element, method, has not always been given proper consideration, given that it is interdependent with the goals to be pursued, which activities are performed, their sequence, and their inter-relationships. Because the language and tool are relatively method independent, it is methodology which further differentiates the effectiveness of any MBSE approach and its ability to help manage the complex and interrelated functionality of today's Cyber Physical Systems. For the approach discussed in this paper, the "methodology" includes not only process, but more significantly the very concept of the underlying information those processes produce and consume, independent of modeling language and tools.

PATTERN-BASED SYSTEMS ENGINEERING

As a Model-Based Systems Engineering (MBSE) methodology, Pattern-Based Systems Engineering (PBSE) can address 10:1 more complex systems with 10:1 reduction in modeling effort, using people from a 10:1 larger community than the "systems expert" group, producing more consistent and complete models sooner. These dramatic gains are possible because projects using PBSE get a "learning curve jumpstart" from an existing pattern and previous users, rapidly gaining the advantages of its content, and improving the pattern with what is learned, for future users. The major aspects of PBSE have been defined and practiced for many years across a number of enterprises and domains, including ground systems. To increase awareness of the PBSE approach, INCOSE has recently started a Patterns Challenge Team within the OMG/INCOSE MBSE Initiative[5].

The term "pattern" appears repeatedly in the history of design, such as civil architecture[6], software design[7], and systems engineering[8]. These are all similar in the abstract, in that they refer to regularities that repeat, modulo some variable aspects, across different instances in space or time. However, the PBSE methodology referred to by this paper is distinguished from those cases by certain important differences:

1. S*Patterns are Model-Based: We are referring here to patterns represented by formal system models, and specifically those which are re-usable, configurable models based on the underlying S*Metamodel. (By contrast, not all the historical "patterns" noted above are described by MBSE models.)

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

Fundamental to Pattern-Based Systems Engineering is the use of the S*Metamodel (summarized by Figure 1), a relational / object information model used in the Systematica™ Methodology to describe requirements, designs, and other information in S*models such as verification, failure analysis, etc.[9.10.11.12.13.14.15]. A metamodel is a model of other models—a framework or plan governing the models that it describes. These may be represented in SysML™, database tables, or other languages.

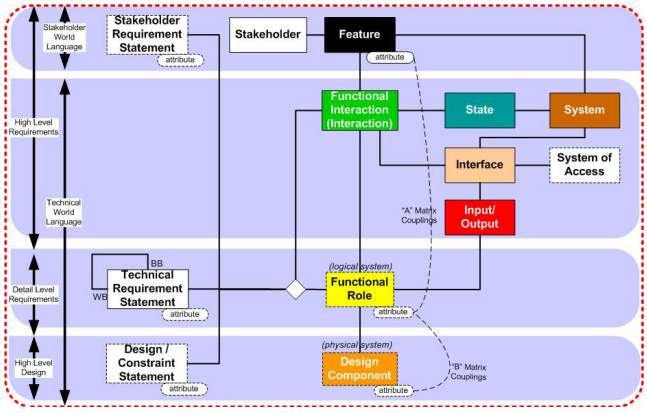

Figure 1: A summary view of the S* metamodel

Specifically, an S*Pattern is a re-usable, configurable S*Model of a family of systems (product line, set, ensemble etc.) as shown in Figure 2 below.

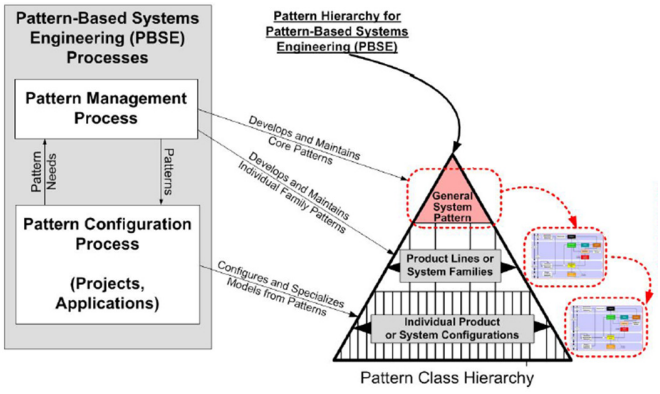

Figure 2: Pattern Hierarchy for PBSE

Over several decades, we have developed and practiced Pattern-Based Systems Engineering across a range of domains, including carrier grade telecommunications, engines and power systems, automotive and off road heavy equipment, telecommunications, military and aerospace, medical devices, pharmaceutical manufacturing, consumer products, and advanced manufacturing systems[16.17.18].

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

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

By contrast, the S*Metamodel is focused on the very physical Interactions that are the basis of the physical sciences, and which we assert are at the heart of the definition of System (in this methodology) as a collection of interacting components[19]. The S*Patterns that arise from the explicit representation of physical Interactions re-form the foundation of system representations to align more explicitly with the physical sciences.

EMERGENCE OF THE EMBEDDED INTELLIGENCE PATTERN FOR CYBER PHYSICAL SYSTEMS

Many S*Patterns are discovered and expressed through PBSE, but the Embedded Intelligence (EI) Pattern is of particular importance to the subject of this paper.

In the world of human-engineered systems, the term "embedded system" has come to be understood to mean a relatively low-level automated control of some sort, typically in electronic hardware/software form, to be inserted into a mechanical or other physical system. In its most common usage, this term is not used to describe larger scale automation, such as would be found for higher-level enterprise information systems or cyber-physical systems. Since cyber-physical systems include both, this sort of divided perspective tends to suggest there are more differences between these levels than we believe are necessary.

Accordingly, the EI Pattern returns to the perspective of Norbert Weiner, who first coined the term "cybernetics" to refer to the study of control and communication in both living and human-engineered systems.[20] This seems particularly appropriate as the term "cyber-physical system" is finding favor in a world in which we seek patterns to advise us at many different hierarchical levels, including systems in which human beings are "embedded".

The EI Pattern is an S*Pattern that emerges to describe intelligence in explicit models of evolving systems in the natural and man-made world—also referred to as the Management System Pattern[21]. It describes the individual elements and overall systemic framework of embedded intelligence on a total system, whether the agents of that intelligence are information technology, human, hybrid, or other forms of management.

Norbert Wiener studied the mathematics of feedback control systems in nature generally, and engineered fire control systems in particular. Although classical feedback control quickly comes to mind in the management of system performance, more generally there are also other aspects of management. The four types of Embedded Intelligence Pattern functional roles that arise are shown graphically in Figure 3:

- Managed System (MDS): Any system behavior whose performance, configuration, faults, security, or accounting are to be managed--referred to as System Management Functional Areas (SMFAs) or in ISO terminology fault, configuration, accounting, performance, security (FCAPS). These are the roles played by the so-called "physical systems" in a cyber-physical system, providing physical services such as energy conversion, transport, transformation, or otherwise.

- Management System (MTS): The roles of performing management (active or passive) of any of the SMFAs of the managed system. These are so-called "cyber" roles in a cyber-physical system, and may be played by automation technology, human beings, or hybrids thereof, to accomplish regulatory or other management purposes.

- System of Users (SOU): The roles played by a system which consumes the services of an managed system and/or management system, including human system users or other service-consuming systems at higher levels.

- System of Access (SOA): The roles providing a means of interaction between the other EI roles. Engineered sensors, actuators, the Internet, and human-machine interfaces have contributed greatly to the emergence of the "Internet of Things".

In evolving systems, instances of these roles may arise individually, and over time lead to an emergent web of embedded intelligence. As further shown in Figure 3, these roles are organized into EI Hierarchies, in which localized EI functions contribute to subsystem intelligence and effectively higher level EI functions contribute to higher level intelligence. In engineered systems, such EI hierarchies may be found in automotive, manufacturing, or aerospace systems, for example.

Like all S*Patterns, the EI Pattern also includes a pattern of Stakeholder Features, Functional Interactions, States, and other aspects which appear repeatedly in a pattern of relationships characteristic of embedded intelligence, whether through planned engineering or emergent evolution. These are not necessarily planned "top down" and the EI Pattern has been used to reverse engineer and describe complex hierarchies of embedded intelligent systems that emerged over time as either human-engineered sub-systems or as natural world systems that include living and other elements.

Within the EI Pattern, States (modes, situations) that arise include Situation Resolution Cycles. As shown in Figure 4, these reflect the idea that system stability over time requires a form of system regulation to "resolve" various "situations" that may occur from time to time, driving the managed system back to a "normal" or nominal state. Examples include:

- Major Mission Resolution Cycles: These proceed through a series of mission states, from mission initiation to fulfilment, including planning.

- Minor Use Case Resolution Cycles: These similarly resolve various situational use cases.

- Resolution of Faults: These may include the recognition, diagnosis, repair, and recovery from system faults.

- Resolution of Service Requests: These may include resolution of requests for such services as re-configuration, security, or other situations.

If a system is capable of not only traveling a situation resolution cycle trajectory (as in Figure 4), but also recognizing that such a situation has arisen in the first place (as in Figure 5), we say that the system is "Situationally Aware".

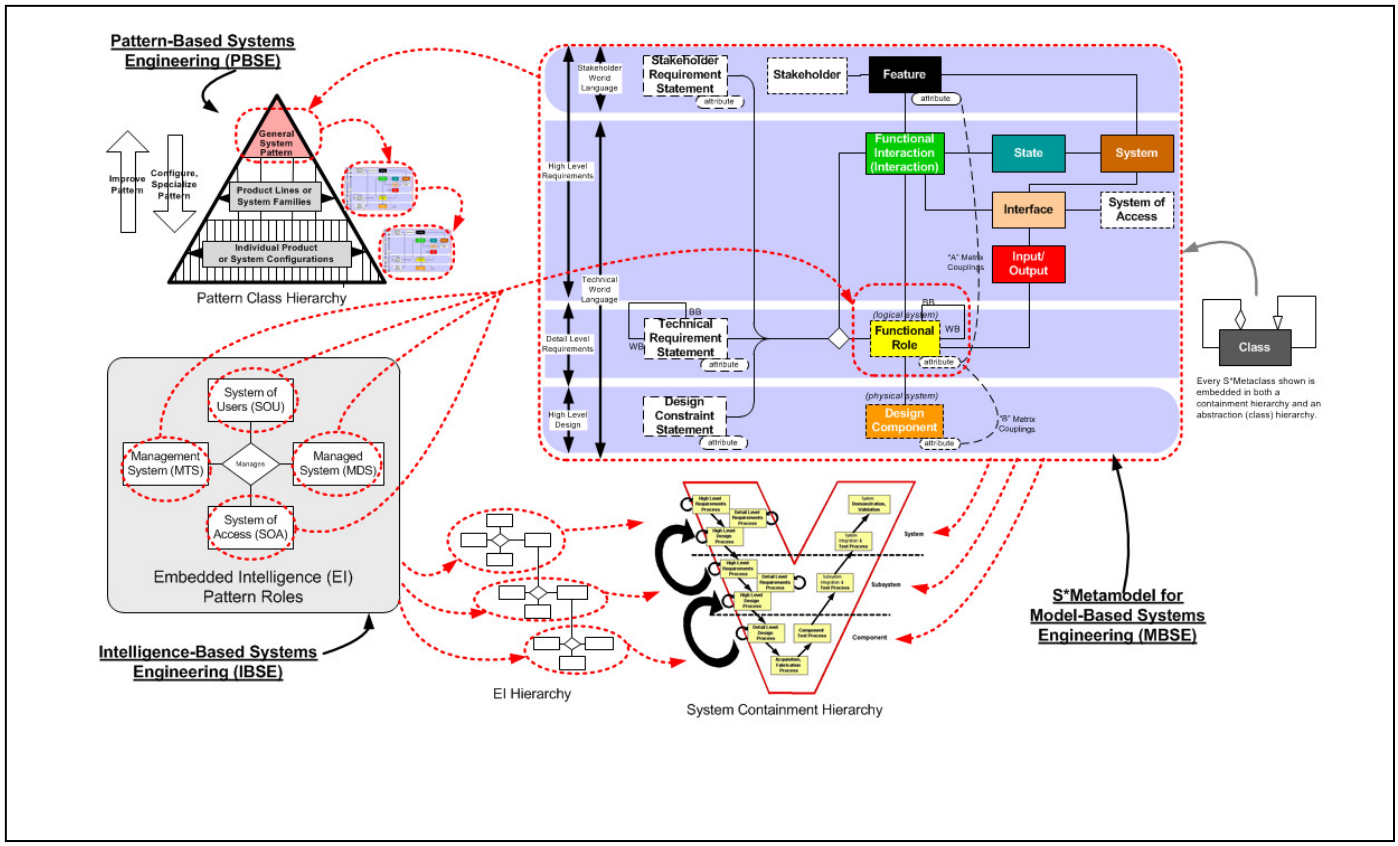

Figure 3: Emergence of Embedded Intelligence (EI) Pattern Functional Roles

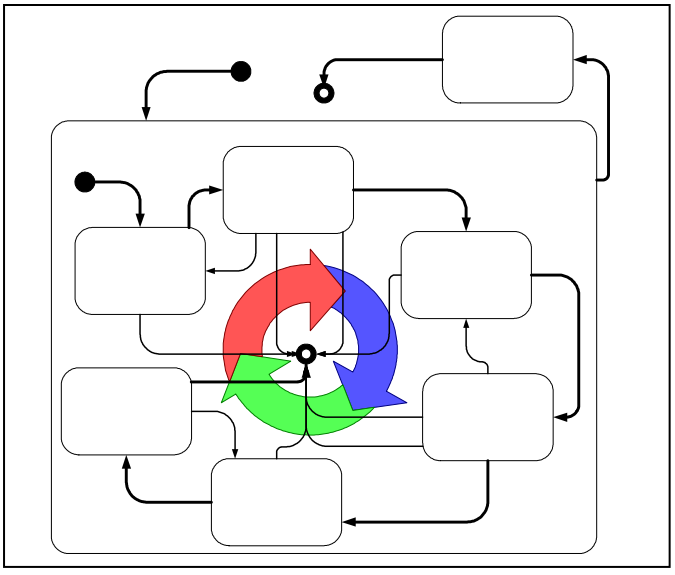

Figure 4: Situation Resolution State Cycles

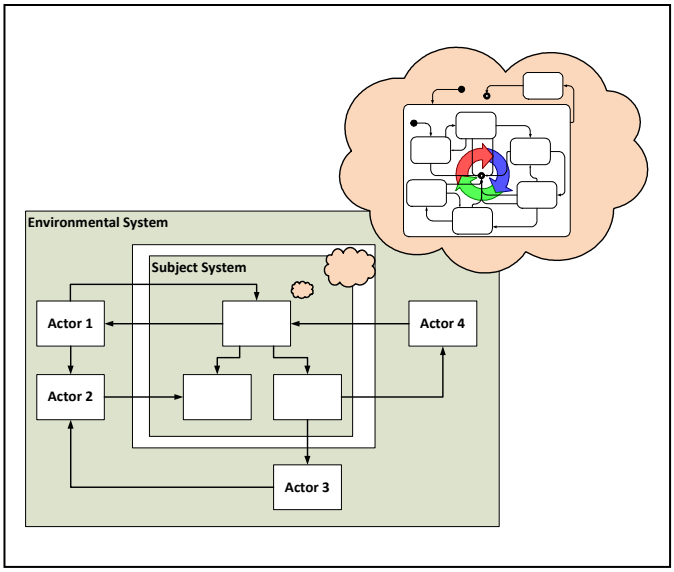

Figure 5: Situational Awareness in Intelligent Systems

If an operator interface panel has a "mode switch", then the system relies on a human operator to recognize occurrence of a situation and place the system in the appropriate state cycle (mode). More advanced autonomous systems automate this (situation recognition and mode-switching) greater sophistication. In all these cases, Attention Models in the EI Pattern describe ability to recognize and resolve situations which arise within the finite limits of managed system and management system resources.

An example of this can be found in ground vehicles in three incremental levels of increasing autonomy – (1) Advanced Driver Assistance Systems (ADAS), (2) Leader-Follower Mode and (3) Full Autonomy. One can imagine the potential architectural differences between systems that have planned to evolve to full autonomy versus those that simply seek to integrate capabilities as they become available. In the second case the full EI pattern is not considered until after a system often experiences architectural lock in and it has limited integration opportunities.

An example of the benefits of applying the EI Pattern to CPS is the insight it provides about emergence of higher level controls and management effects in systems, which we have observed to be frequently overlooked or misunderstood—especially in high complexity systems such as autonomous ground vehicles and the larger domain systems they inhabit. The EI Pattern is based first and foremost on the logical roles of that pattern, not the physical technologies that perform those roles. A common thinking trap is to associate control system boundaries or scope with physical controllers or information system platforms. The EI Pattern teaches us that these boundaries are better understood using the scope of Logical Management System roles. It is commonplace in engineering of control systems or information systems to expend effort in "integrating" those information systems so that they can interact with each other ("talk" to each other). What is often overlooked, and what the EI Pattern enforces in our understanding, is that a higher level management system emerges logically in any situation in which two management system roles that have different managed system scopes interact with each other. This is a powerful way to understand how higher level management effects (whether positive or negative) emerge even if we did not initially recognize them.

CONCLUSIONS

As a model-based systems engineering approach PBSE is particularly well suited to address CPS challenges. PBSE provides a data model and framework that is both holistic and compact. It addresses the core system science needed in designing CPS by making interactions, the heart of Cyber Physical Systems, more visible. PBSE and the EI pattern provides a rapid and holistic means to identify and manage system risk and failure identification, analysis, and planning essential to CPS. Both are also essential in establishing patterns of adaptive and hierarchical control which can be leveraged as a framework for engineering trusted systems. The Embedded Intelligence Pattern explicitly represents the logical roles which enable planned evolution and limits architectural lock in, effectively reducing switching costs and speeding technology integration.

REFERENCES

[1] "NSF-Funded Joint Efforts", at www.nsf.gov/funding/pgm_summ.jsp?pims_id=503286

[2] "The Most Innovative Companies 2013 – Lessons from Leaders" 2013 BCG Global Innovators Survey, BCG Analytics.

[3] INCOSE SE Vision 2020

[4] http://www.omgwiki.org/MBSE/doku.php?id=start

[5] http://www.omgwiki.org/MBSE/doku.php?id=mbse:patterns:patterns

[6] Christopher Alexander, Sara Ishikawa, Murray Silverstein, Max Jacobson, Ingrid Fiksdahl-King, and Shlomo Angel. A Pattern Language. Oxford University Press, New York, 1977.

[7] Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides. Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley Publishing Company, Reading, MA, 1995.

[8] Robert Cloutier. Applicability of Patterns to Architecting Complex Systems: Making Implicit Knowledge Explicit. VDM Verlag Dr. Müller. 2008.

[9] Bill Schindel, Troy Peterson, "Introduction to Pattern-Based Systems Engineering (PBSE): Leveraging MBSE Techniques", in Proc. of INCOSE 2013 Great Lakes Regional Conference on Systems Engineering, Tutorial, October, 2013.

[10] W. Schindel, "System Interactions: Making The Heart of Systems More Visible", in Proc. of INCOSE Great Lakes 2013 Regional Conference on Systems Engineering, October, 2013.

[11] Abbreviated Systematica Glossary, Ordered by Concept, V 4.2.2, ICTT System Sciences, 2013.

[12] W. Schindel, "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.

[13] W. Schindel, "Failure Analysis: Insights from Model-Based Systems Engineering", in Proceedings of INCOSE 2010 Symposium, July 2010.

[14] W. Schindel, "Pattern-Based Systems Engineering: An Extension of Model-Based SE", INCOSE IS2005 Tutorial TIES 4, (2005).

[15] W. Schindel, "Requirements Statements Are Transfer Functions: An Insight from Model-Based Systems Engineering", in Proc. of INCOSE 2005 International Symposium, (2005).

[16] W. Schindel, and V. Smith, "Results of Applying a Families-of-Systems Approach to Systems Engineering of Product Line Families", SAE International, Technical Report 2002-01-3086 (2002).

[17] J. Bradley, M. Hughes, and W. Schindel, "Optimizing Delivery of Global Pharmaceutical Packaging Solutions, Using Systems Engineering Patterns", in Proc. of the INCOSE 2010 International Symposium (2010).

[18] W. Schindel, "Integrating Materials, Process & Product Portfolios: Lessons from Pattern-Based Systems Engineering", in Proc. of 2012 Conference of Society for the Advancement of Material and Process Engineering, 2012.

[19] W. Schindel, "What Is the Smallest Model of a System?", in Proc. of the INCOSE 2011 International Symposium, International Council on Systems Engineering (2011).

[20] Norbert Weiner, Cybernetics: Control and Communication in the Animal and the Machine, Cambridge, MA, MIT Press, 1965.

[21] W. Schindel, and V. Smith, "Results of Applying a Families-of-Systems Approach to Systems Engineering of Product Line Families", SAE International, Technical Report 2002-01-3086 (2002).