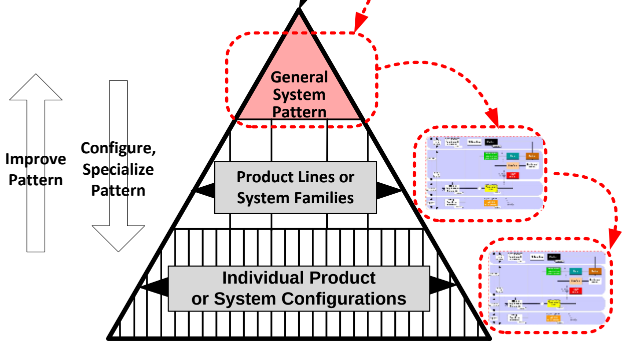
A joint IW2016 workshop by:

- INCOSE Patterns Working Group
- INCOSE Product Line Engineering Working Group

INCOSE 2016 international workshop
Los Angeles, CA, USA
January 30 - February 2, 2016

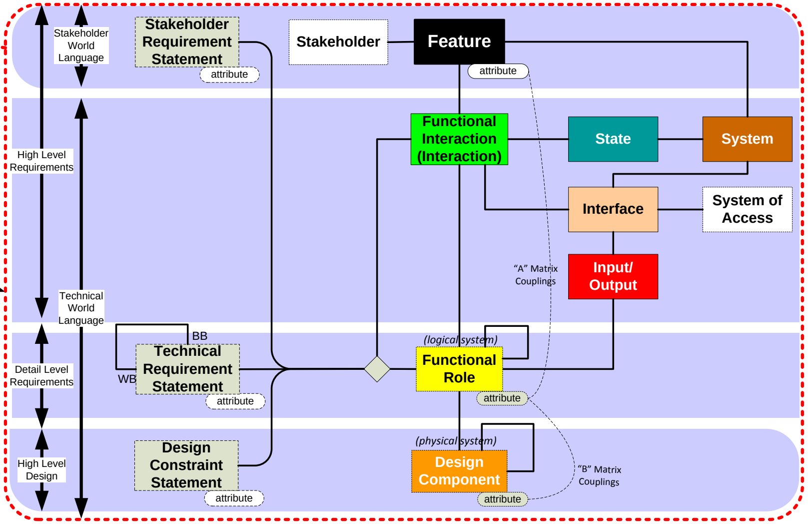
Extracting PLE Patterns for Legacy Systems

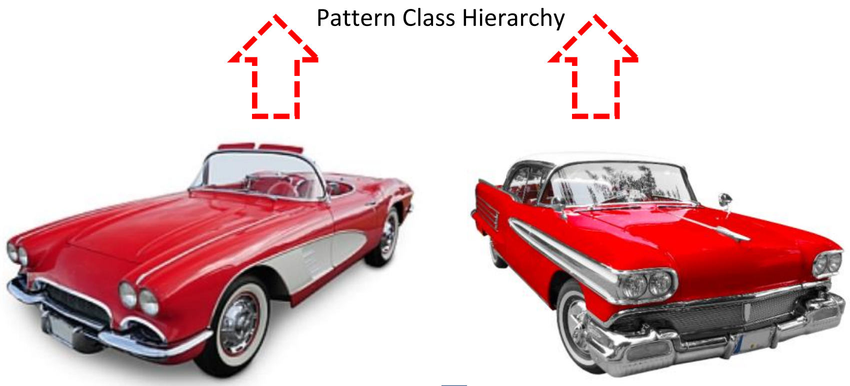
Pattern Hierarchy for Pattern-Based Systems Engineering (PBSE)

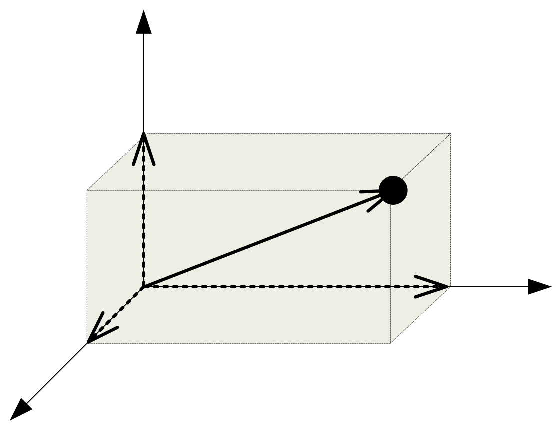


Pattern Class Hierarchy

Metamodel for Model-Based Systems Engineering (MBSE)







V1.5.2

Contents

- Workshop motivation, background, and objectives
- Workshop agenda / time line

- Introduction to PLE: Convergence of interests, leading to interest in legacy and other PLE patterns
- Introduction to the Method of Projections for generating PLE MBSE Patterns from and for Legacy Systems
- Discussion of ideas and interests by the PLE WG and the Patterns WG membership in potential joint projects
- Wrap up

- References

Motivation:

- Product Line Engineering (PLE) is increasingly recognized as a major improver of business outcomes, with advancing methods, tools, and standards.
- Pattern-Based Systems Engineering (PBSE) provides powerful means to create, evolve, and apply MBSE Patterns as configurable models, including Product Lines.

Working Group Background:

- The INCOSE PLE Working Group is several years into exploration of Product Line Engineering, including principles and examples[1,2].
- The INCOSE Patterns Working Group is several years into exploration of S*Patterns, representing MBSE models of systems across domains, using the S*Metamodel[3,4].

Objectives of this Workshop:

- This is a "mini" workshop summarizing one aspect, intended to measure interest in more substantial future joint activities by the two working groups:
  - We will briefly introduce the Method of Projections, a means of generating PLE MBSE Patterns from and for legacy systems.
- A discussion by the attendees will be used to gauge interest in future pursuit of this or other subjects and projects of mutual interest.

1. http://www.biglever.biz/extras/OOPSLA06_LSI.pdf
2. http://www.incose.org/docs/default-source/wgcharters/product-lines.pdf?sfvrsn=6
3. http://www.omgwiki.org/MBSE/doku.php?id=mbse:pbse
4. http://www.omgwiki.org/MBSE/doku.php?id=mbse:patterns:patterns

Workshop Agenda and Time Line

Workshop Session	Time	Lead
Patterns WG business start up (before joint workshop)	*13:00 – 13:20*	*Patterns WG*
Introduction, review of joint workshop objectives and agenda	13:20 – 13:35	Joint
Introduction to the PLE WG: Convergence of interests, leading to interest in legacy and other PLE patterns	13:35 – 13:55	PLE WG
Introduction to the Method of Projections for generating PLE MBSE Patterns from legacy systems	13:55 – 14:35	Patterns WG
Discussion of ideas and interests by the PLE WG and Patterns WG membership in potential joint projects	14:35 – 14:55	Attendees
Wrap up	14:55 – 15:00	Joint
IW-Wide Break	*15:00 - 15:30*	

Introduction to the PLE WG:

Convergence of interests, leading to interest in legacy and other PLE patterns

PLE Working Group

Product Line Engineering Intl. WG

- Purpose
  - To promote PLE and related SE best practices
    - Coordinate activities around PLE at INCOSE level and share results
- Goals
  - Help our members acquire Know-How
    - Compare to the State-of-Art
    - Share concerns, experiences, good practices and traps to avoid
  - Provide guidelines to setup and evolve PLE in organizations
- Scope
  - All types of Systems, Markets & Organizations
  - All the SE Processes (needs, requirements, architecture, integration and tests …)
  - All maturity levels of PLE, from opportunistic to completely Integrated and anticipated strategies

Product Line Engineering Intl. WG

Steering Committee

- **Chairs/co-chairs**: Hugo Guillermo Chalé Góngora / Charlie Krueger (INCOSE Central) / Alain Le Put (AFIS WG)
- **Work products**: Bob Malone (Boeing)
- **ISO committee rep.:** Charlie Krueger (BigLever)
- **Outreach**: Paul Clements (BigLever)
- **INCOSE Liaison**: Matthew Hause-Trasnport/MBSE (PTC), Bill Bolander-Automotive (IBM), Jean-Claude Roussel-Requirements (Airbus), Jim Hummel-Tool interoperability (PTC), Anil Prasad-Healthcare (Medtronic plc)
- **Events**: TBD
- **Webinars**: Suresh Tirumalai (GE Oil&Gas), Konstantinos Vilaetis (NY AirBrake)
- **IT & Services**: Barclay Brown (IBM), Karen Smiley (ABB Group)

Number of Members: ~85

What is a Product Line?

A product line is a set of products (system) that share a **common**, managed set of characteristics and that are developed from a common set of core assets in a prescribed way

- The product line satisfies the specific needs of a particular market segment or of an ensemble of segments

The products of a product line present **variable** characteristics that differentiate them from one another

Reuse and PLE

- Reuse is related to concepts like platform engineering, product family engineering or PLE

- PLE defines a process to manage the underlying architectures of the product platforms (or portfolio) of an organization in order to **maximize the benefits of reuse**
  - "Architectures": all kinds of structured, organized data used to characterize our systems in their entirety

- **Reuse** should be the result of a well-documented decision process → Implementing PLE requires upfront investment and thought

Key concepts of PLE

- One of the main challenges in PLE consist in identifying or defining the “set of **core** assets” of the product line
  - The non-varying, “stable” characteristics that will be reused “of-the-shelf” on all products
- For companies with a considerable amount of (inconsistent) legacy assets, a simple question naturally arises
  - How to formalize and leverage these to help the to define a product line?

PLE and Patterns from afar

- Patterns (boiler plates, templates…) are good candidate tools to formalize reusable items
- They might also provide the tools to “extract” reusable assets from legacy assets

This was the original motivation of this workshop

Introduction to the Method of Projections

for generating PLE MBSE Patterns from and for Legacy Systems

MBSE Patterns Working Group

Introduction to the Method of Projections

- Method of Projections – Procedural Overview
- What are S*Models?
- What are S*Patterns?
- Examples, past and future

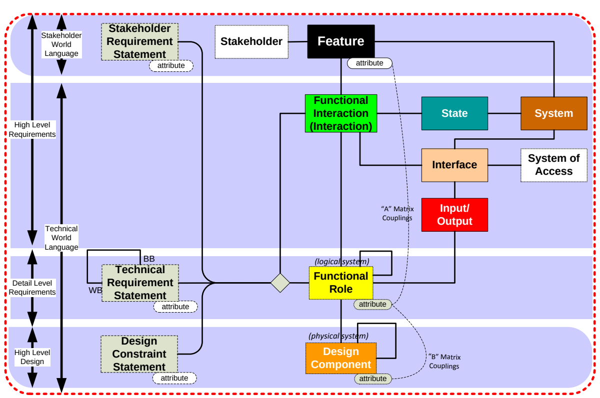
Method of Projections: Procedural Overview

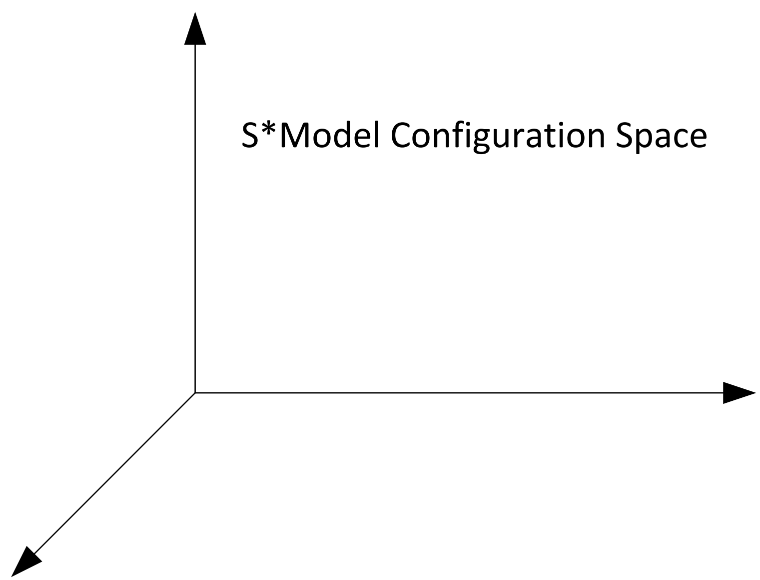
1. Identify sources of Legacy Configuration information (partial, informal, the system itself, etc.) about the legacy system(s).
2. Identify an "initial guess" draft S*Pattern as a starting point—may be very incomplete, or mis-matched at first, or a portfolio parent pattern.
3. For each incremental chunk of the Legacy Configuration information:
   a) Carry out Projection Procedure of that part of the Legacy Configuration onto the Draft Pattern, effectively re-expressing it in the Draft Pattern MBSE language.
   b) Identify projection overshoots and undershoots compared to the Pattern.
   c) Analyze needed refinements to the Draft Pattern.
   d) Perform incremental adjustments to Draft Pattern.
4. Perform a trial configuration of the Draft Pattern, to re-generate a configuration of the Legacy System:
   a) Compare the resulting configuration to the Legacy System.
   b) Check internal configuration consistency (e.g., Requirements-Design)
   c) Depending upon differences, repeat 3-4 if necessary.

(Although simple in principle, this is actually the PBSE form of "the loop of science".)

The Method of Projections

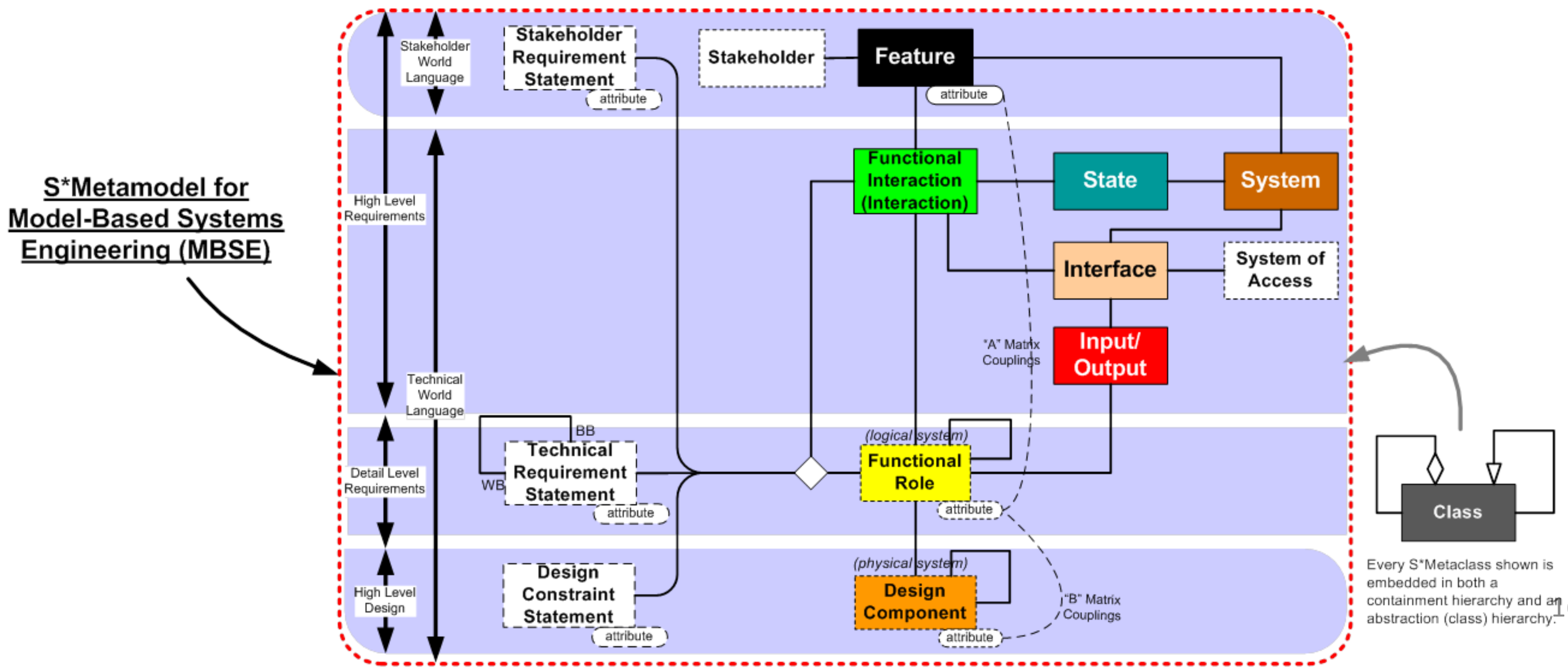
- Takes advantage of being <u>model-based</u>:
  - Using the idea of System Configuration Space.
  - The configuration space axes (degrees of model configuration freedom) are determined by the S*Pattern.
  - Those degrees of freedom are expressed through the S*Metamodel.





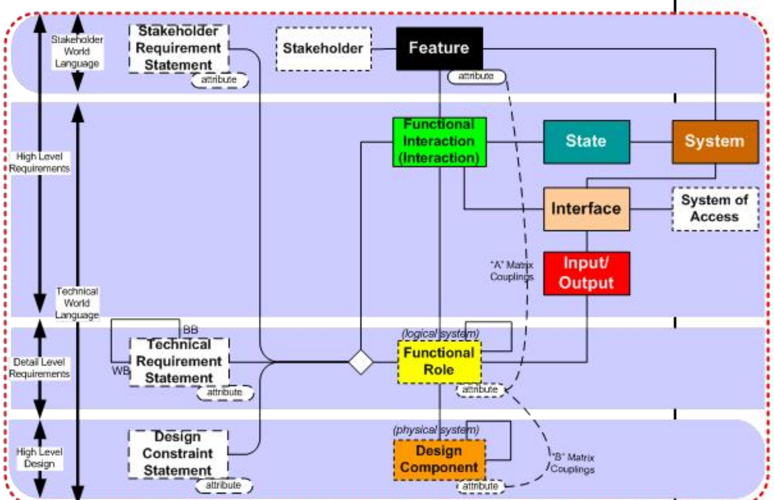
What are S*Models?

- S*Models are MBSE system models that are based on the S*Metamodel:
  - Independent of specific modeling language.
  - S*Metamodel maps into any contemporary modeling language, including OMG SysML®, third party COTS tools.



What are S*Models?

- S*Models are MBSE models that are based on the S*Metamodel:
  - The smallest amount of modeled information necessary for purposes of science or engineering.

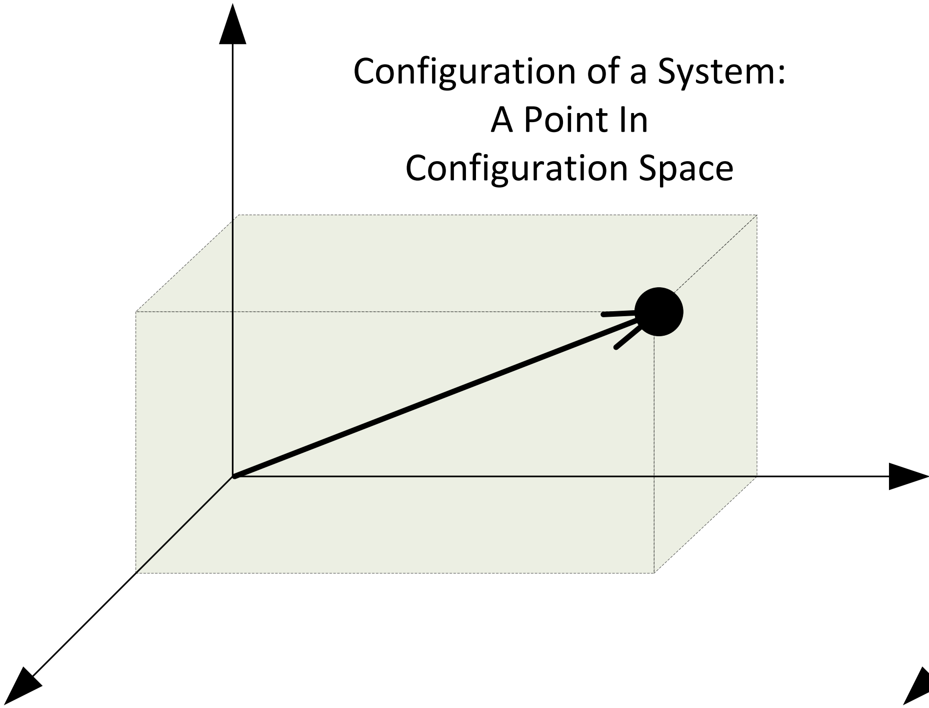


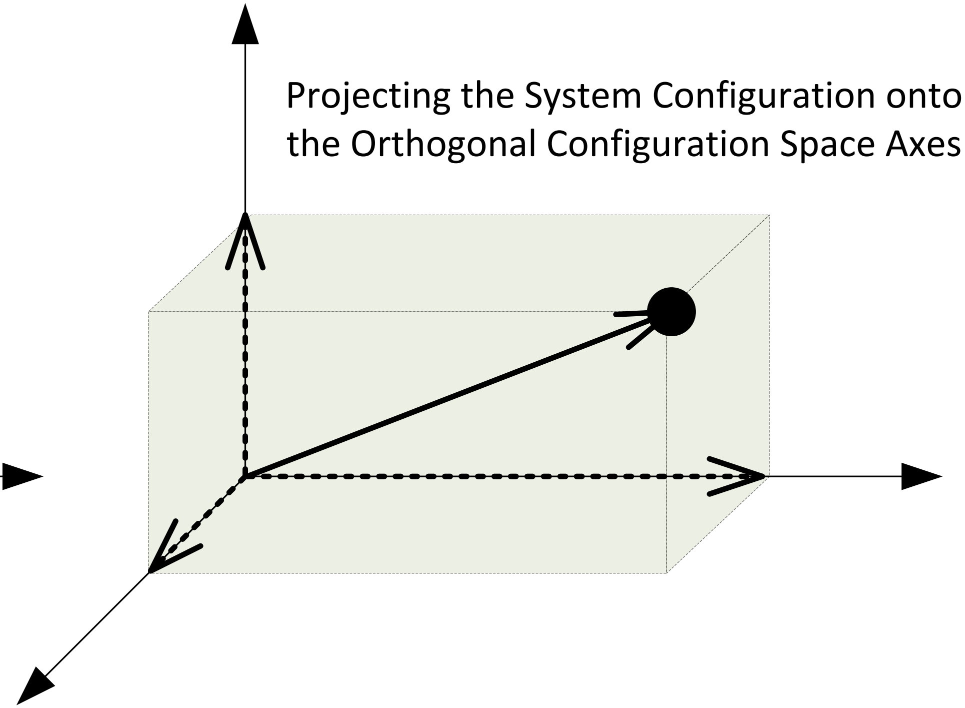
What Is the Smallest Model of a System?

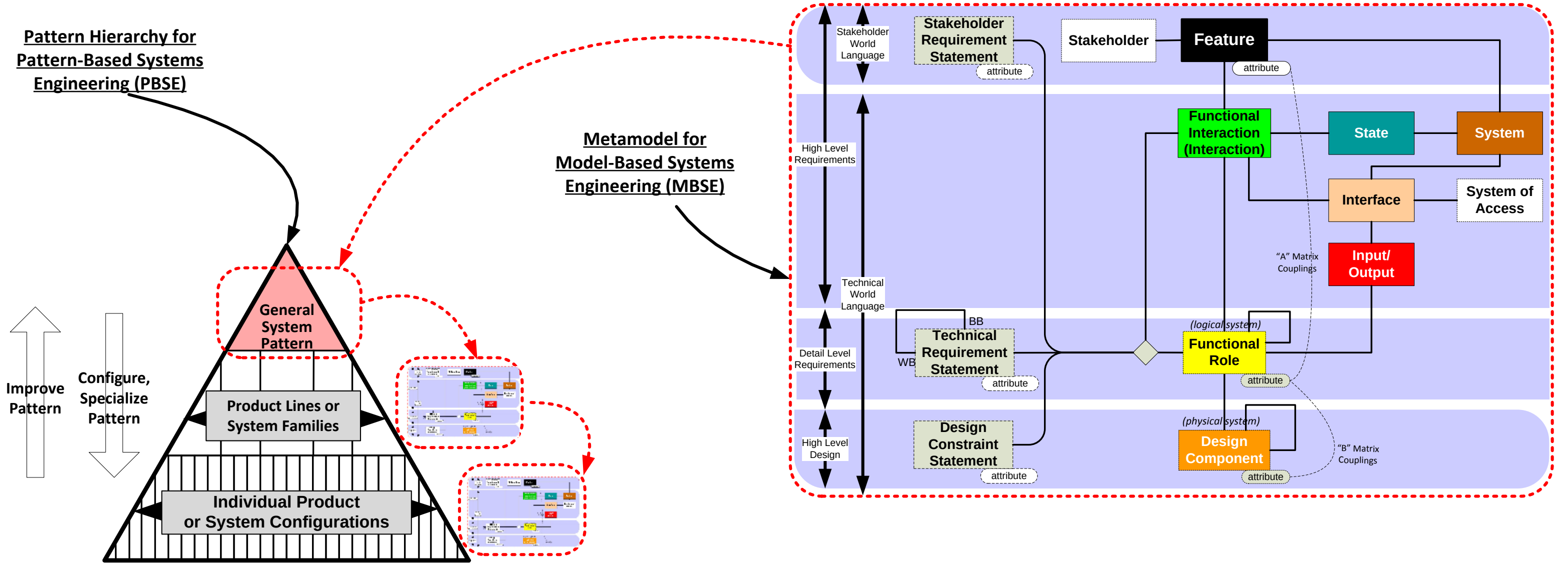
William D. Schindel
ICTT System Sciences
schindel@ictt.com

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

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



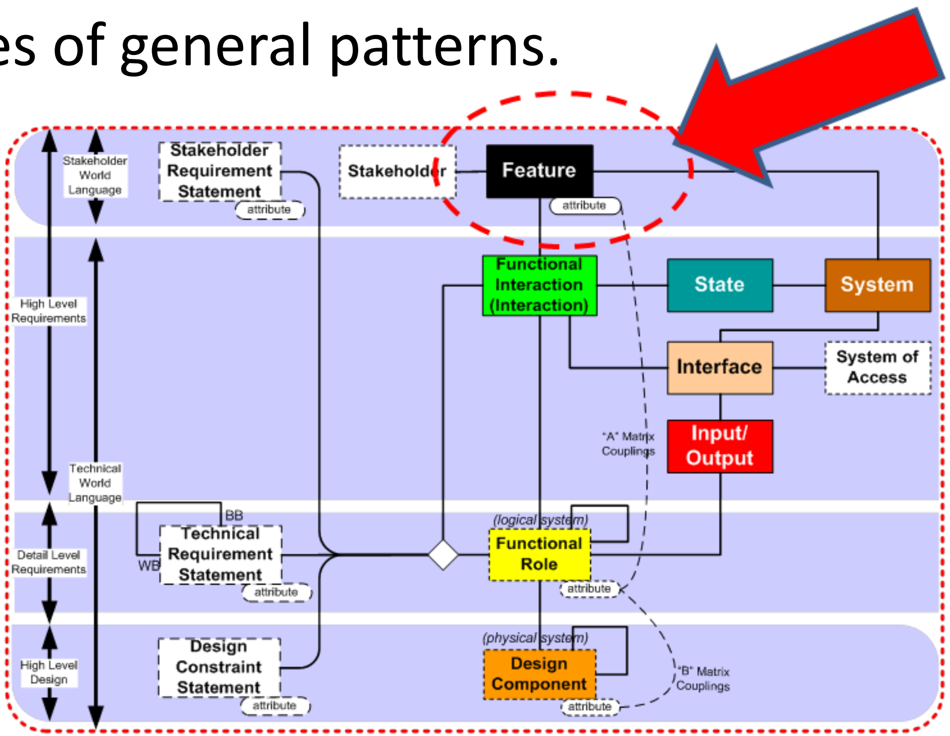




Pattern Class Hierarchy

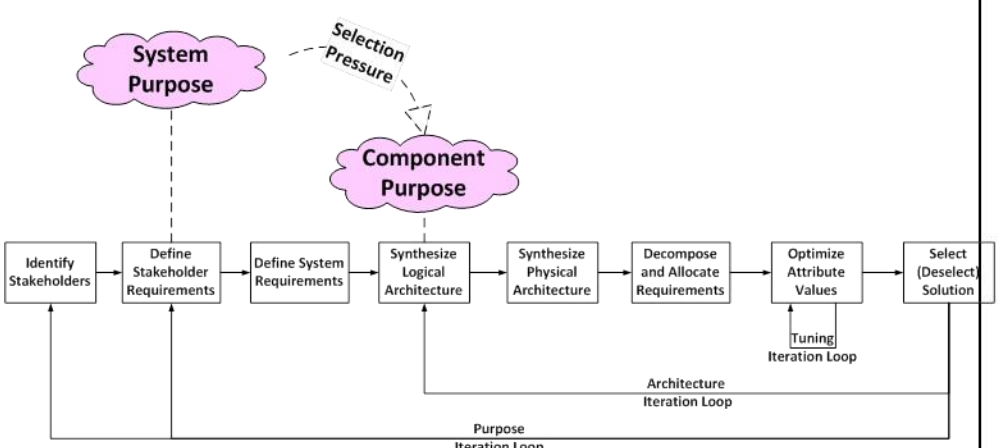
Some Elements of S*Models

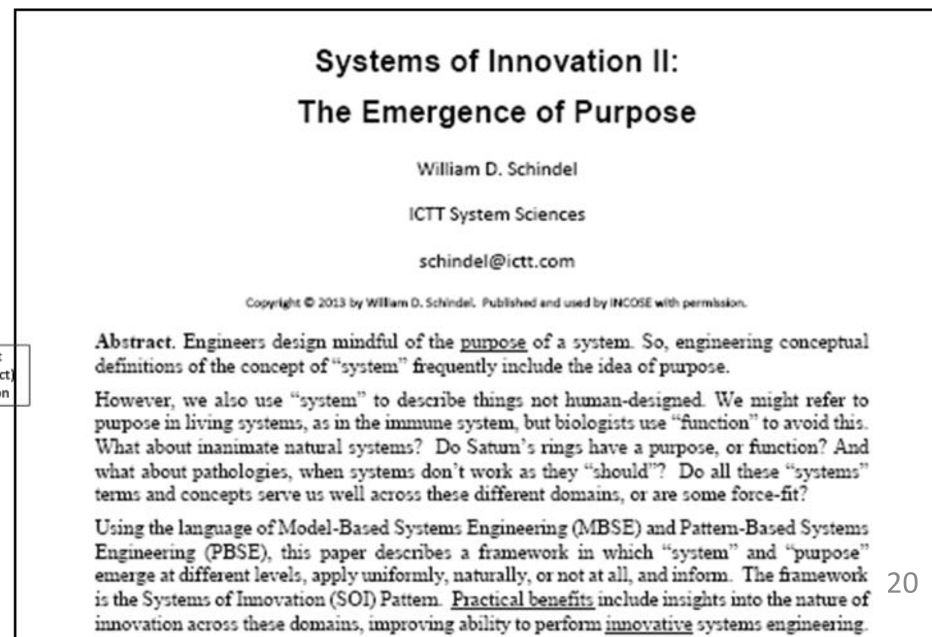
- **Features** express emergent, selectable value (fitness) as expressed by selection mechanisms (market, cognitive, biological, other):
  - When we want to represent fitness, goodness of performance, or other expressions of value in system product lines, then Features, parameterized by Feature Attributes, model that fitness space.
  - Features also provide a natural basis for driving configuration of specific cases of general patterns.



Some Elements of S*Models

- **<u>Features</u>** express emergent, selectable value (fitness) as expressed by selection mechanisms (market, cognitive, biological, other):
  - The <u>purpose</u> of a system is the functional role for which it is selected, or the role it performs in a (larger) selected system.
  - System purpose (function in biology) emerges over time, even in human-designed systems.
  - PLE Patterns require a way to express configurable fitness, value.





Systems of Innovation II: The Emergence of Purpose

William D. Schindel

ICTT System Sciences

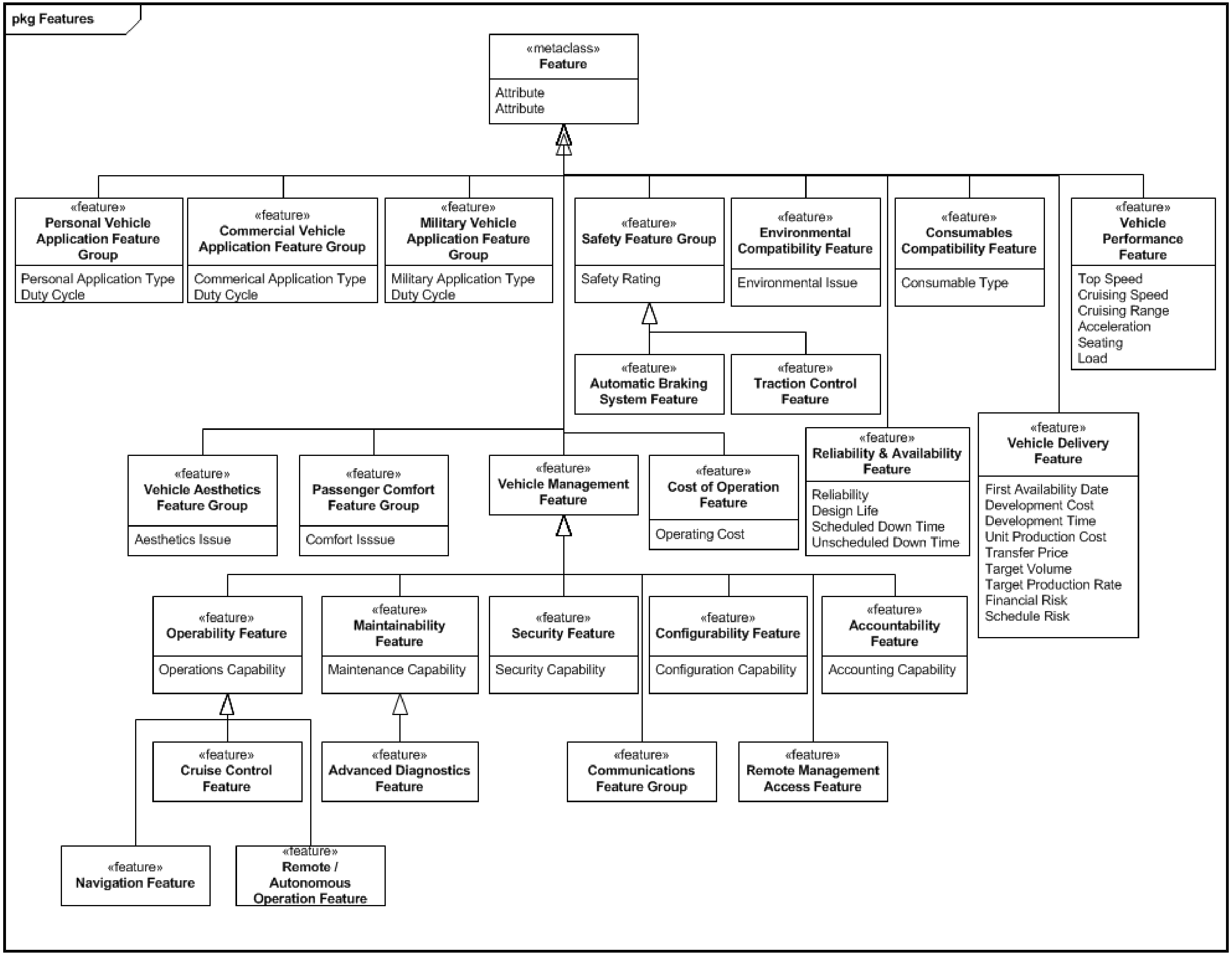
schindel@ictt.com

Abstract. Engineers design mindful of the <u>purpose</u> of a system. So, engineering conceptual definitions of the concept of "system" frequently include the idea of purpose.

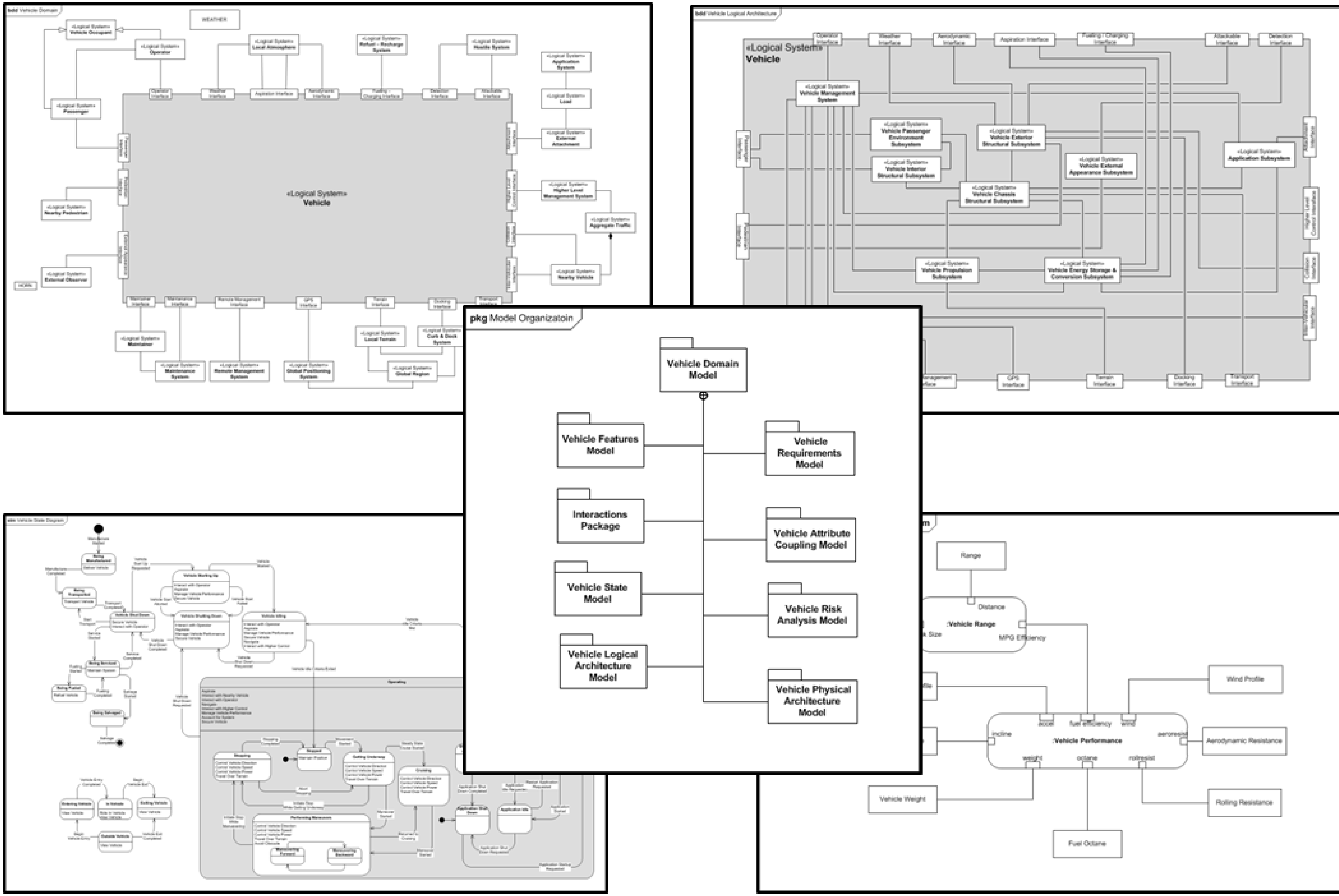
However, we also use "system" to describe things not human-designed. We might refer to purpose in living systems, as in the immune system, but biologists use "function" to avoid this. What about inanimate natural systems? Do Saturn's rings have a purpose, or function? And what about pathologies, when systems don't work as they "should"? Do all these "systems" terms and concepts serve us well across these different domains, or are some force-fit?

Using the language of Model-Based Systems Engineering (MBSE) and Pattern-Based Systems Engineering (PBSE), this paper describes a framework in which "system" and "purpose" emerge at different levels, apply uniformly, naturally, or not at all, and inform. The framework is the Systems of Innovation (SOI) Pattern. <u>Practical benefits</u> include insights into the nature of innovation across these domains, improving ability to perform <u>innovative</u> systems engineering.

Vehicle Pattern Feature Package

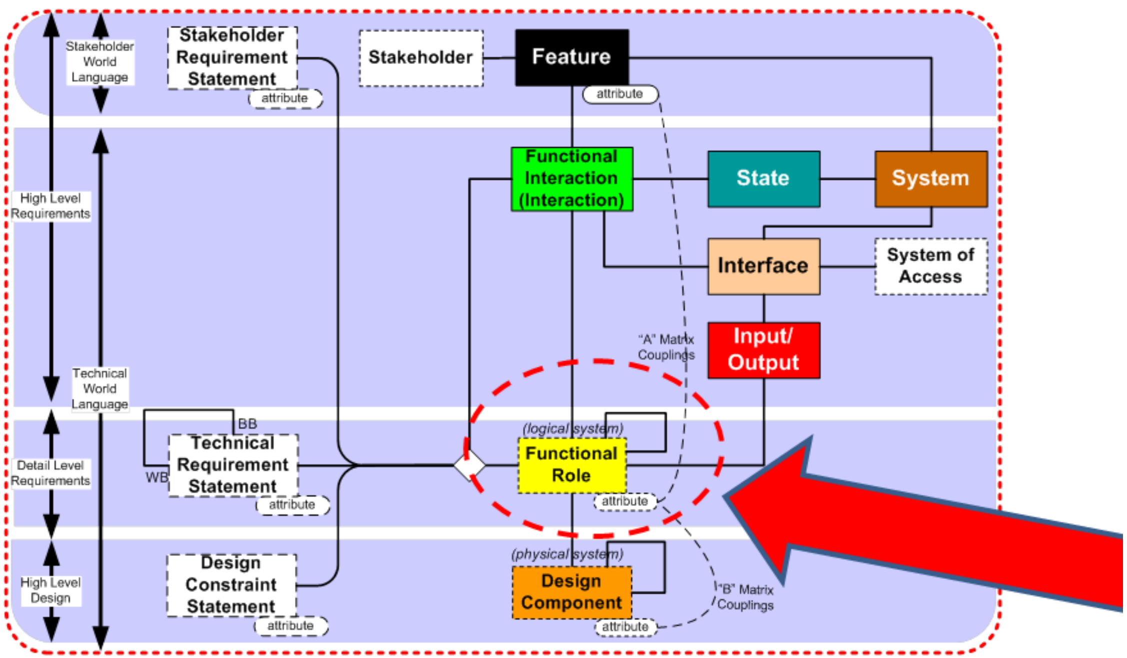


Additional Vehicle Pattern Packages



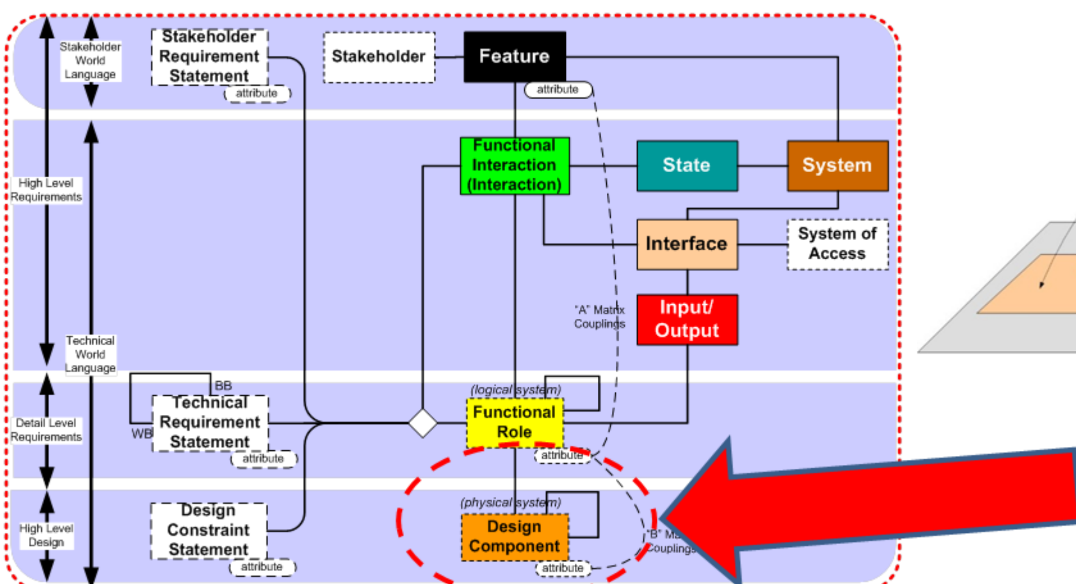
Some Elements of S*Models

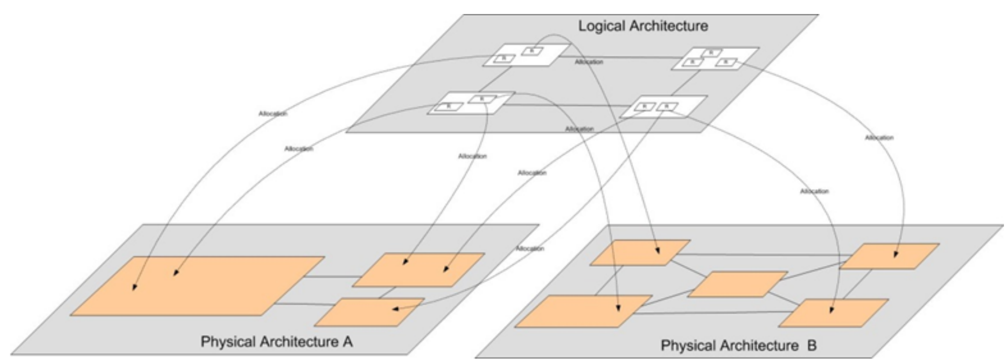
- **Functional Roles**: Describe chunks of behavior, independent of the physical things that perform it, parameterized by role Attributes.
- **Architectural Relationships**: These connect Functional Roles, to describe Logical Architecture
- PLE patterns require a means of expressing logical architecture patterns, and its configuration for specific cases.



Some Elements of S*Models

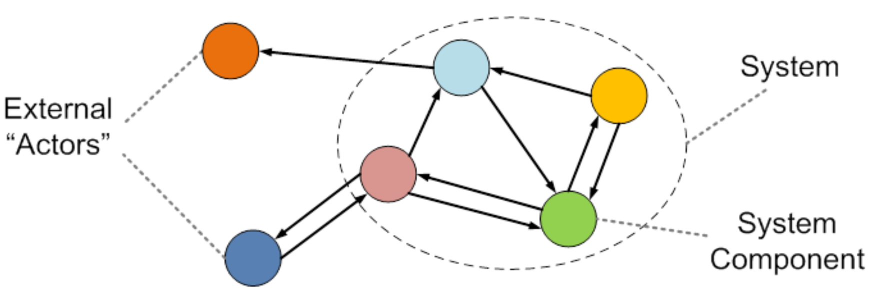
- **<u>Design Components</u>**: Model component identities, without behavior, and parameterized by Attributes.
- **<u>Architectural Relationships</u>**: Connect Design Components, to describe Physical Architecture.
- **<u>Allocation Relationships</u>**: Describe allocations of Functional Roles to Design Components.
- PLE patterns offer configurable allocations to different physical architectures.

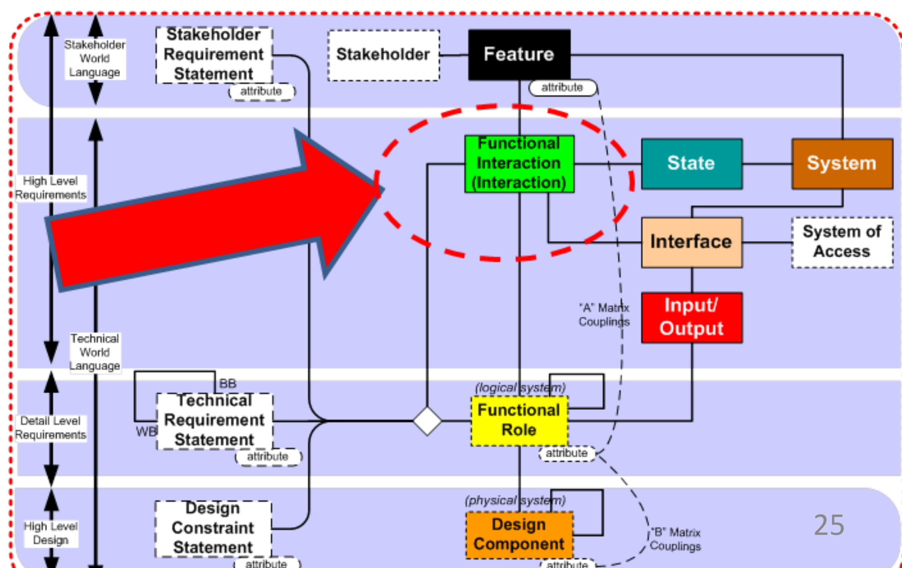




Some Elements of S*Models

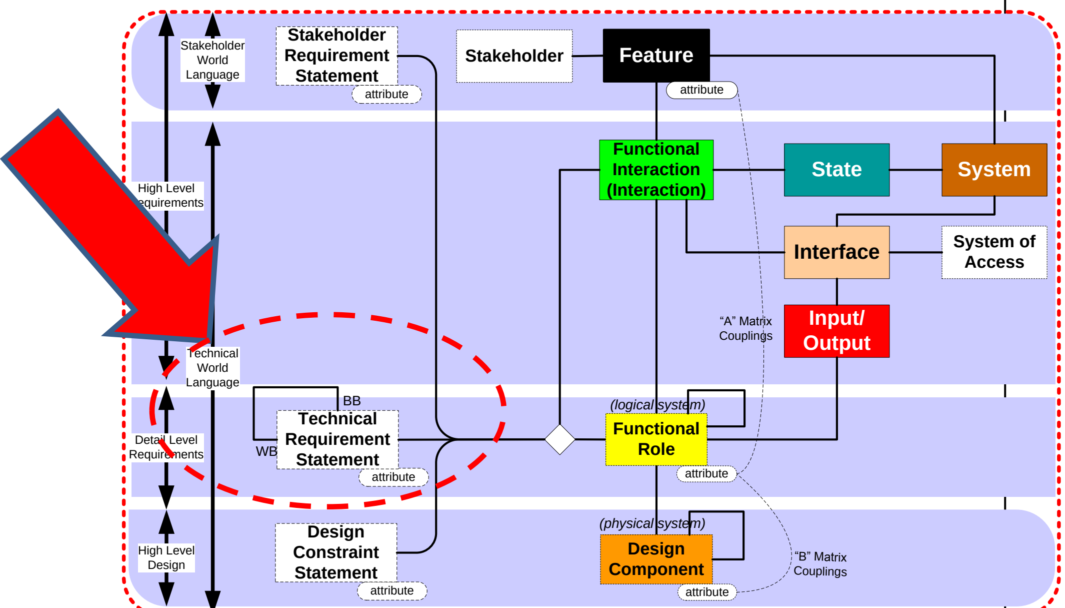
- **<u>Interactions</u>** are at the heart of the S*Metamodel, and SE.
- This approach defines a <u>System</u> as a collection of interacting components:
  - By “interact” we mean exchanges of force, energy, mass, or information, resulting in changes of state.
  - Virtually all the laws of the physical sciences uncovered during the last 300 years are expressed in terms of such Interactions.
  - All behavior occurs as interacting Functional Roles.
  - Interactions are central to SE.

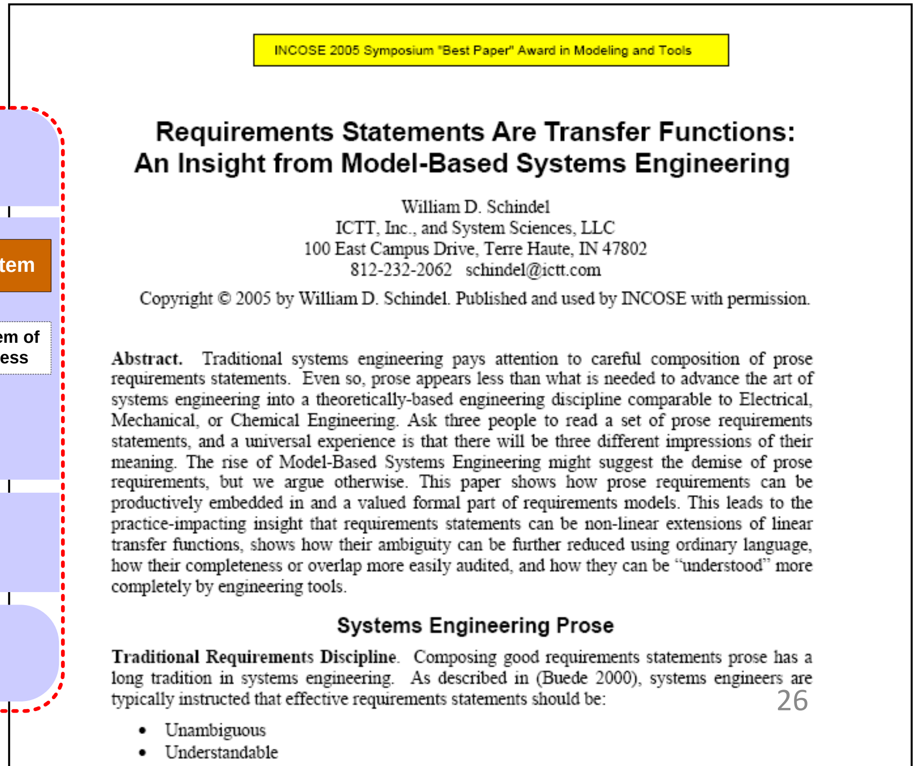




Some Elements of S*Models

- **Requirements Statements** describe modeled behavior of Functional Roles during Interactions:
  - In traditional prose or other forms, but always describing input-output relationships.
  - Parameterized by Requirements Attributes.
  - Variable configurable in PLE patterns.





INCOSE 2005 Symposium "Best Paper" Award in Modeling and Tools

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

William D. Schindel
ICTT, Inc., and System Sciences, LLC
100 East Campus Drive, Terre Haute, IN 47802
812-232-2062 schindel@ictt.com

Copyright © 2005 by William D. Schindel. Published and used by INCOSE with permission.

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

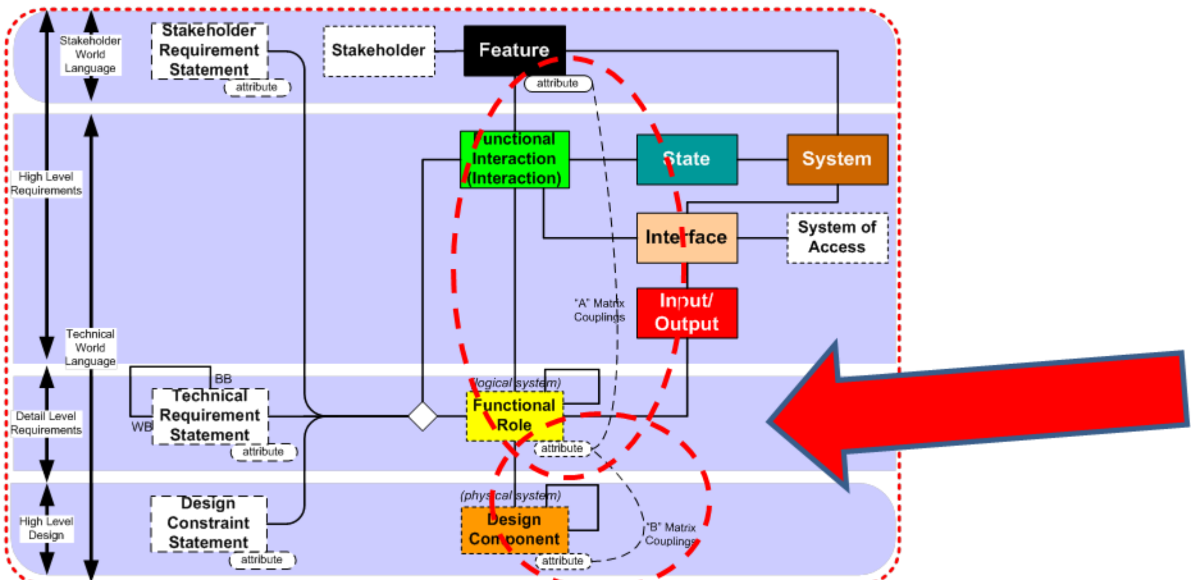
Systems Engineering Prose

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

- Unambiguous
- Understandable

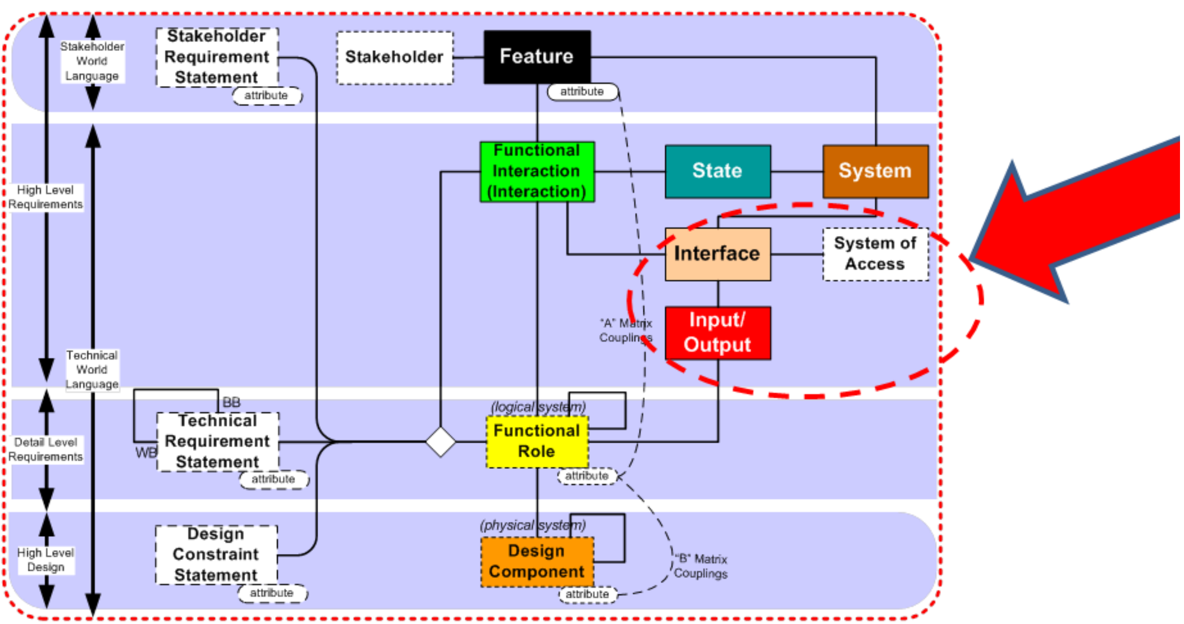
Some Elements of S*Models

- **<u>Attribute Couplings</u>** identify quantitative relationships between quantitative attributes (parametric relationships):
  - <u>A Couplings</u>: Express how fitness or value is coupled to technical behavior.
  - <u>B Couplings</u>: Express how technical behavior is coupled to chosen components.
- We are interested in representing what we can extract about these couplings for Legacy PLE Patterns.



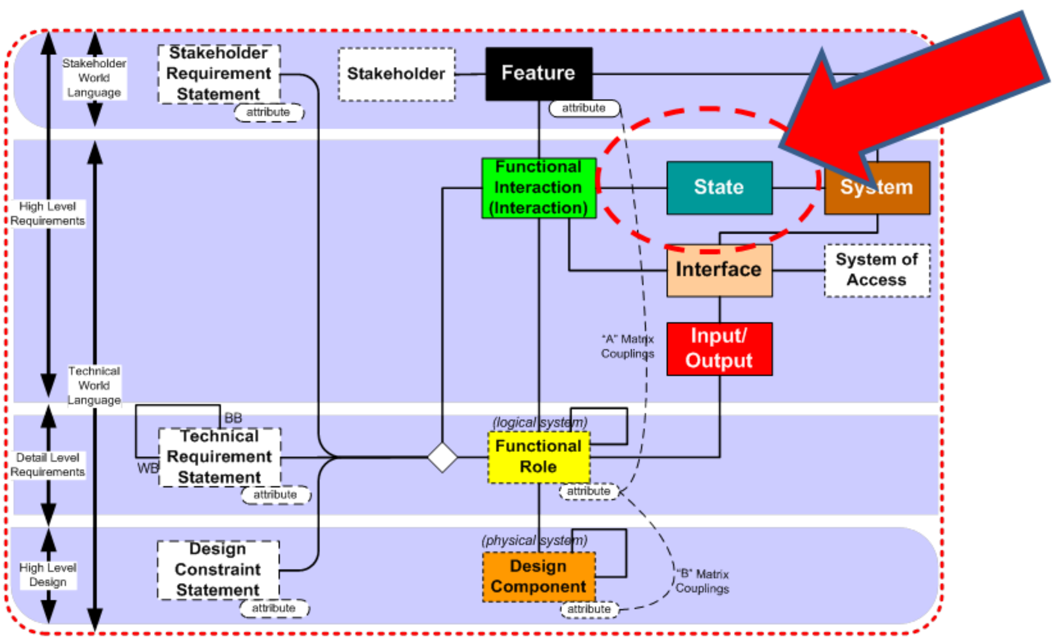
Some Elements of S*Models

- **<u>Interfaces</u>** describe behavior at system boundaries or between components:
  - What Interfaces exist?
  - What **<u>Input-Outputs</u>** are exchanged at an Interface?
  - What is the behavior at an Interface? (Interactions)
  - What is the **<u>System of Access</u>** at an Interface?
- We are interested in modeling interfaces for Legacy PLE Patterns.



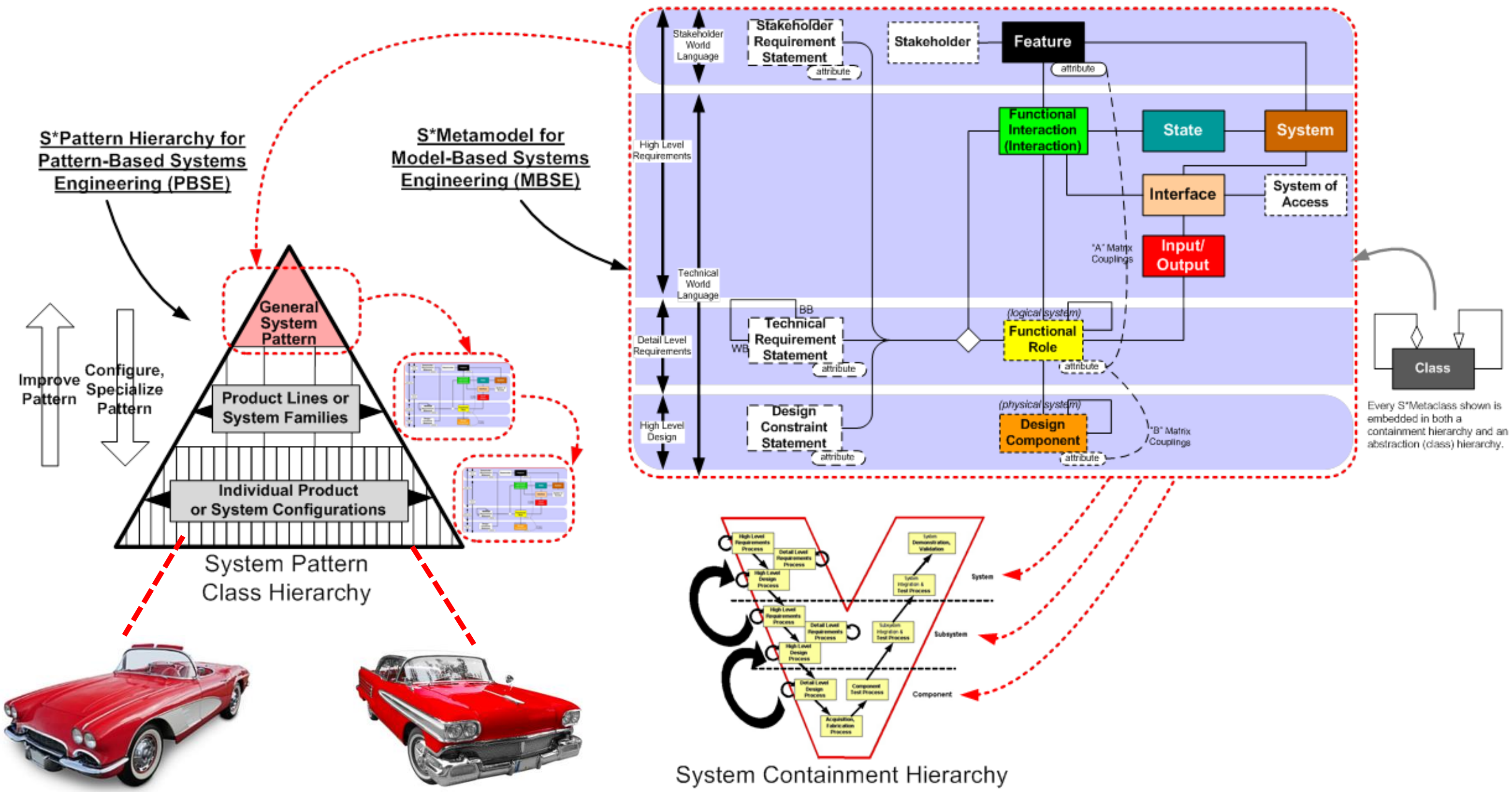
Some Elements of S*Models

- **<u>States, Modes, and Phases</u>** describe conditions or situations of systems:
  - In different system states, system behavior may be different, by intent or nature, or need to be different.
  - We are interested in the states of a Legacy System, its environment, or its subsystems.



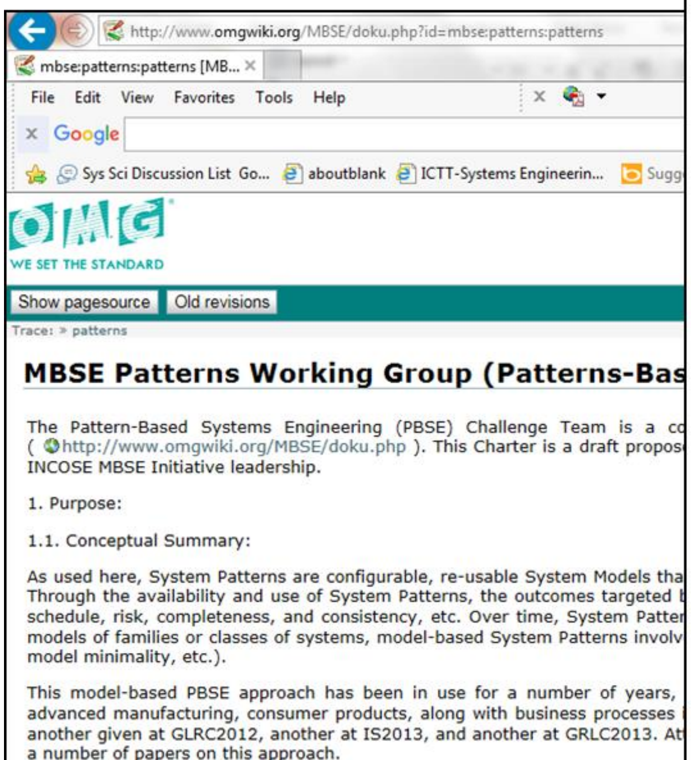
What are S*Patterns?

- S*Patterns are configurable, re-usable S*Models of families of systems:
  - Architectural Frameworks, Product Lines, Platforms, etc.
  - A form of model compression.
  - Using the elements of the S*Metamodel.



What are S*Patterns?

- The basis of Pattern-Based Systems Engineering (PBSE), an extension of MBSE:
  - The focus of the INCOSE Patterns Working Group, an INCOSE/OMG MBSE Initiative Challenge Team



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

mbse:patterns:patterns [MB...

File Edit View Favorites Tools Help

Google

Sys Sci Discussion List Go... aboutblank ICTT-Systems Engineerin... Sugg

OMG WE SET THE STANDARD

Show pagesource Old revisions

Trace: » patterns

MBSE Patterns Working Group (Patterns-Bas

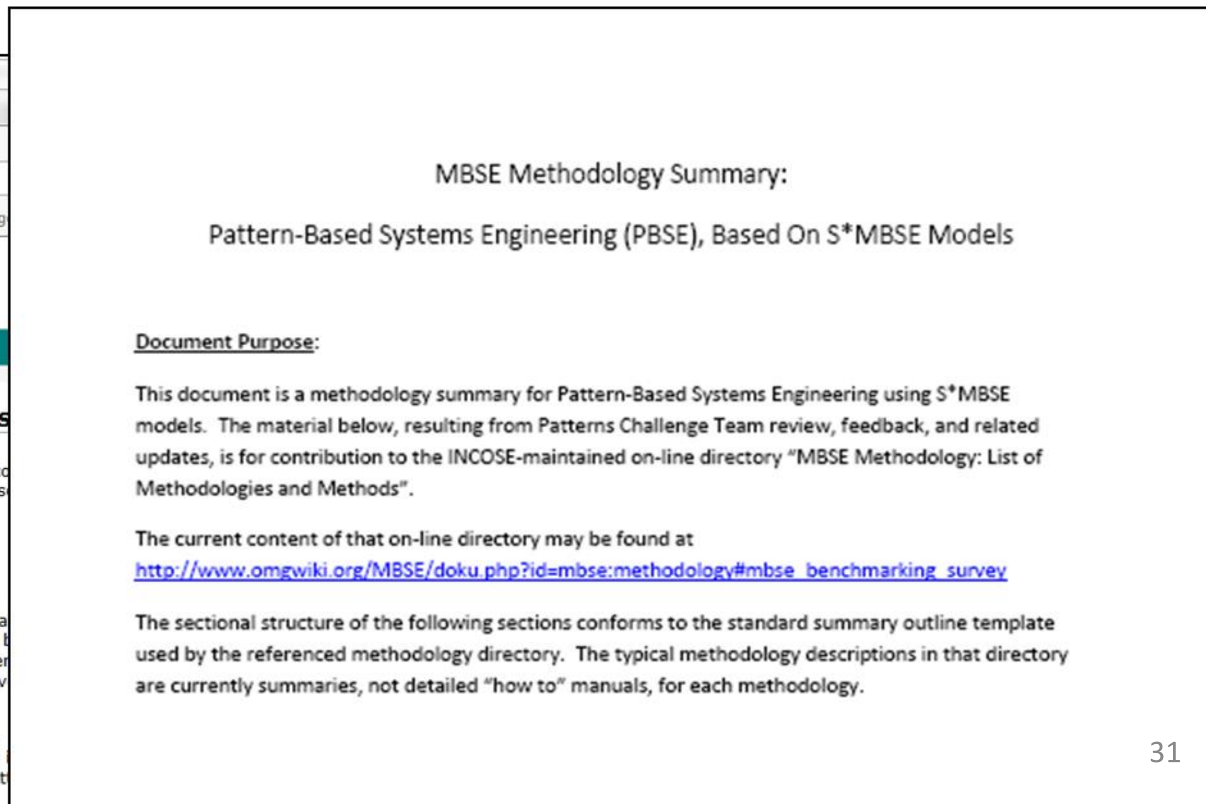
The Pattern-Based Systems Engineering (PBSE) Challenge Team is a co (http://www.omgwiki.org/MBSE/doku.php). This Charter is a draft propos INCOSE MBSE Initiative leadership.

1. Purpose:

1.1. Conceptual Summary:

As used here, System Patterns are configurable, re-usable System Models tha Through the availability and use of System Patterns, the outcomes targeted b schedule, risk, completeness, and consistency, etc. Over time, System Patter models of families or classes of systems, model-based System Patterns involv model minimality, etc.).

This model-based PBSE approach has been in use for a number of years, advanced manufacturing, consumer products, along with business processes another given at GLRC2012, another at IS2013, and another at GRLC2013. At a number of papers on this approach.



MBSE Methodology Summary:

Pattern-Based Systems Engineering (PBSE), Based On S*MBSE Models

Document Purpose:

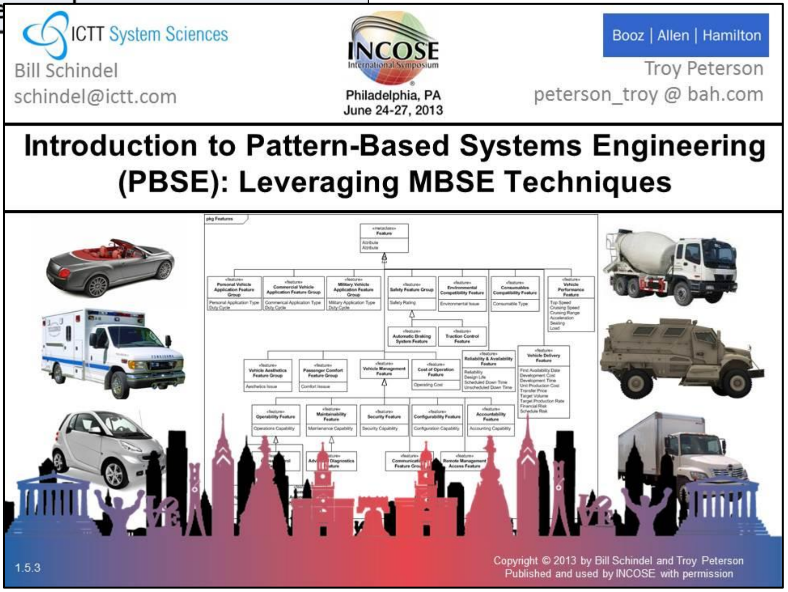
This document is a methodology summary for Pattern-Based Systems Engineering using S*MBSE models. The material below, resulting from Patterns Challenge Team review, feedback, and related updates, is for contribution to the INCOSE-maintained on-line directory "MBSE Methodology: List of Methodologies and Methods".

The current content of that on-line directory may be found at http://www.omgwiki.org/MBSE/doku.php?id=mbse:methodology#mbse_benchmarking_survey

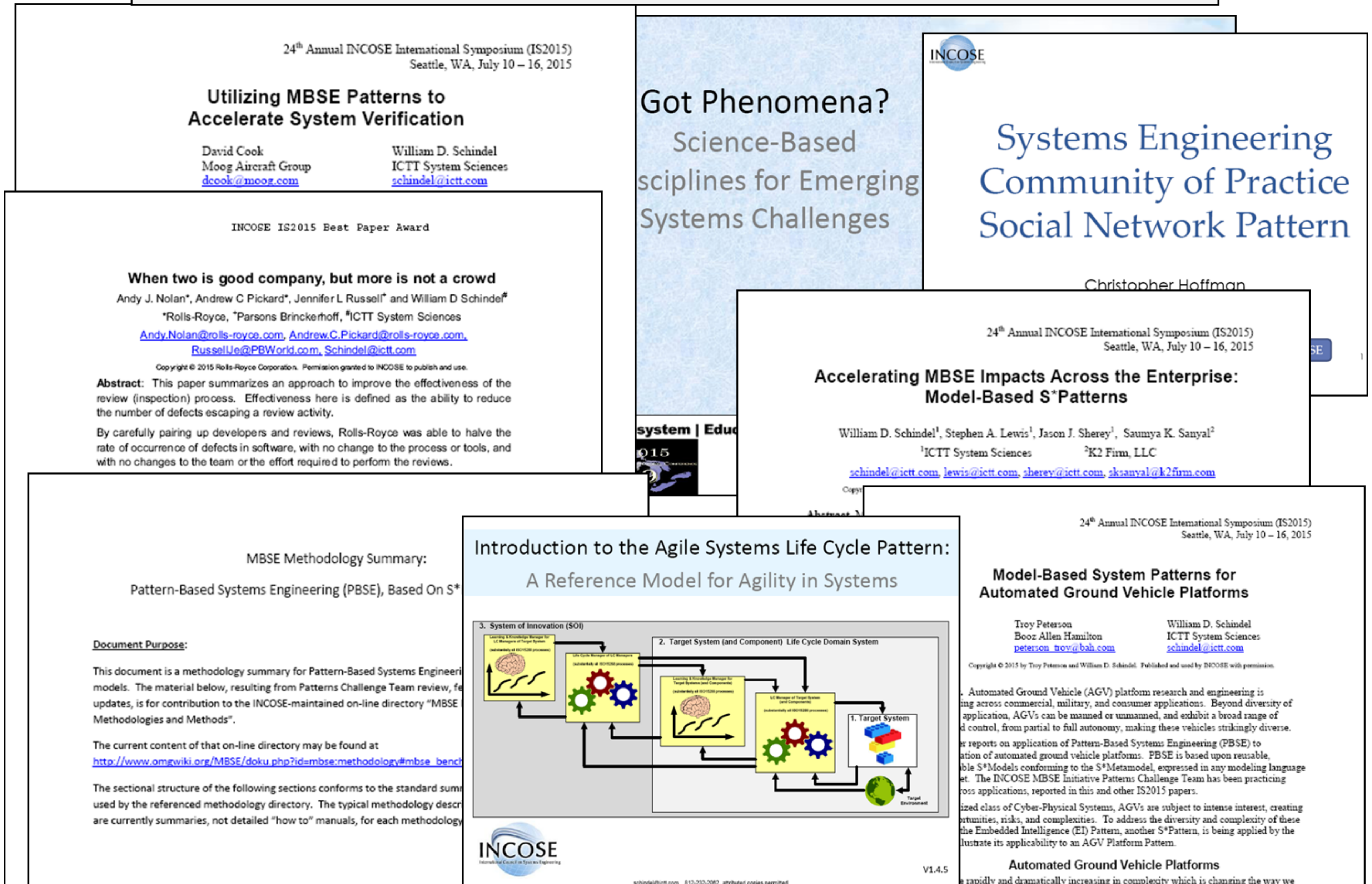
The sectional structure of the following sections conforms to the standard summary outline template used by the referenced methodology directory. The typical methodology descriptions in that directory are currently summaries, not detailed "how to" manuals, for each methodology.

S*Patterns have been applied across many domains, over several decades

Medical Devices Patterns	Construction Equipment Patterns	Commercial Vehicle Patterns	Space Tourism Pattern
Manufacturing Process Patterns	Vision System Patterns	Packaging Systems Patterns	Lawnmower Product Line Pattern
Embedded Intelligence Patterns	Systems of Innovation (SOI) Pattern	Consumer Packaged Goods Patterns (Multiple)	Orbital Satellite Pattern
Product Service System Patterns	Product Distribution System Patterns	Plant Operations & Maintenance System Patterns	Oil Filter Pattern
Life Cycle Management System Patterns	Production Material Handling Patterns	Engine Controls Patterns	Military Radio Systems Pattern
Agile Systems Engineering Life Cycle Pattern	Transmission Systems Pattern	Precision Parts Production, Sales, and Engineering Patte	Higher Education Experiential Pattern

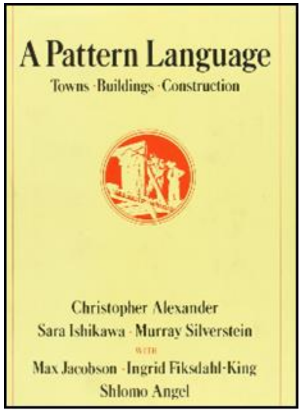


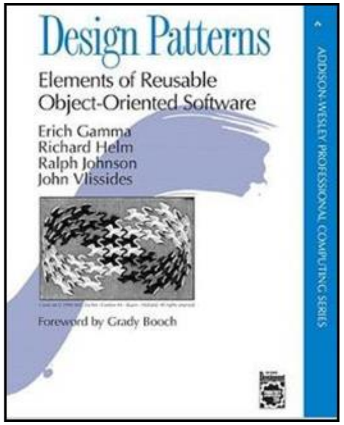
Some Pattern-Related Publications by Patterns Working Group Members

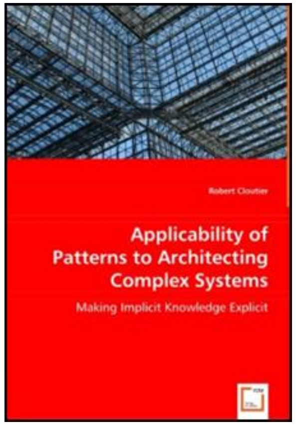


What are S*Patterns?

- The term “pattern” appears repeatedly in the history of design, such as civil architecture, software design, and systems engineering:







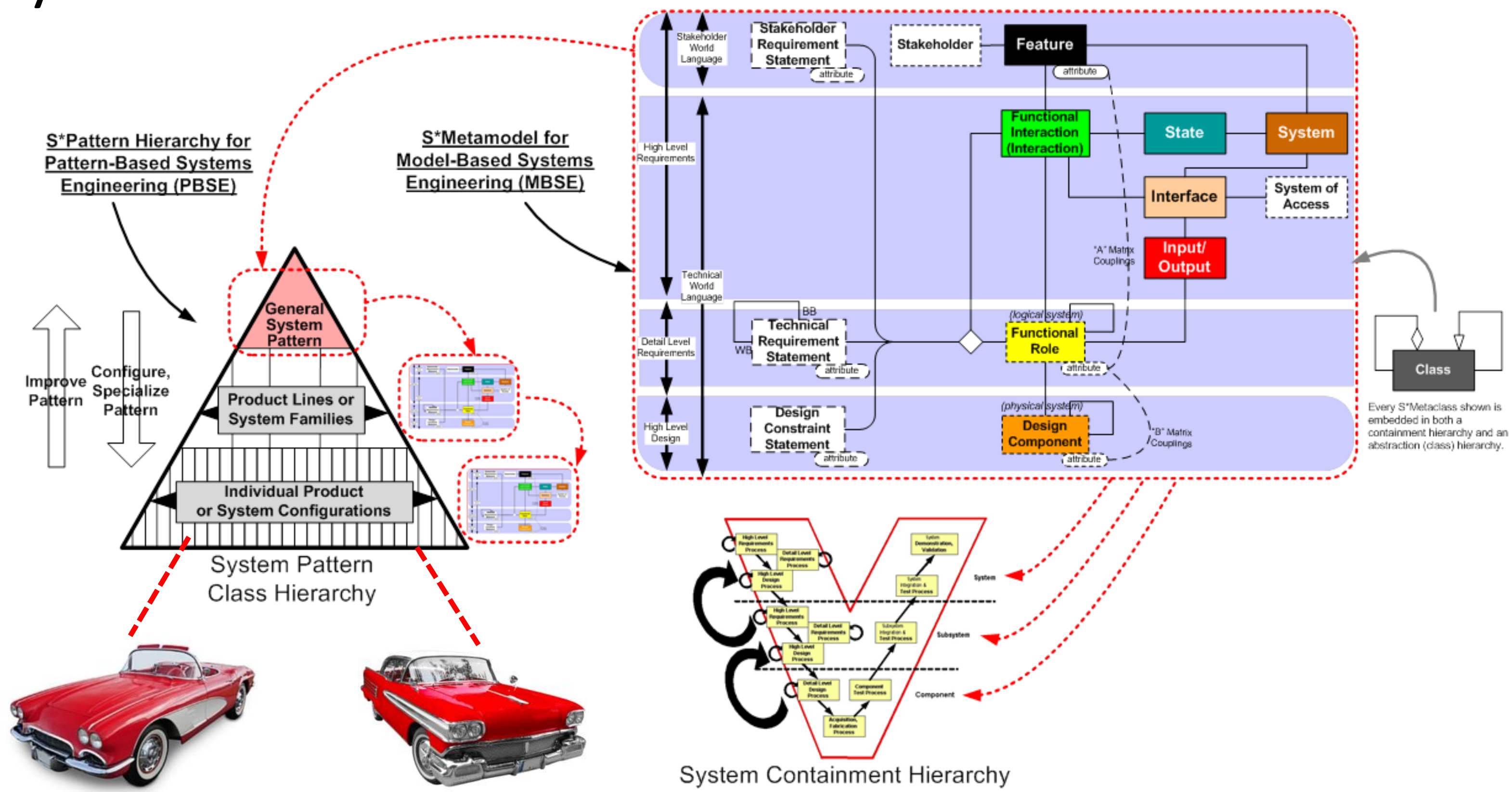
- Those “patterns” represent regularities that repeat, modulo some variable aspects, across different instances in space, time, and other dimensions.
- However, when we refer to “patterns” in the Patterns WG, we mean the use of <u>S*Patterns</u>.

What are S*Patterns?

- S*Patterns are <u>model-based</u> (not all historical "patterns" are expressed as MBSE models).
- S*Patterns conform to the S*Metamodel—as a minimal reference model of essential engineering information.
- S*Patterns are embedded in modeled concepts about <u>physical interactions</u> that are the basis of physical laws of the hard sciences emerging over the last 300 years.
- S*Patterns are about "<u>whole systems</u>" (historical "patterns" were sometimes about parts of systems).

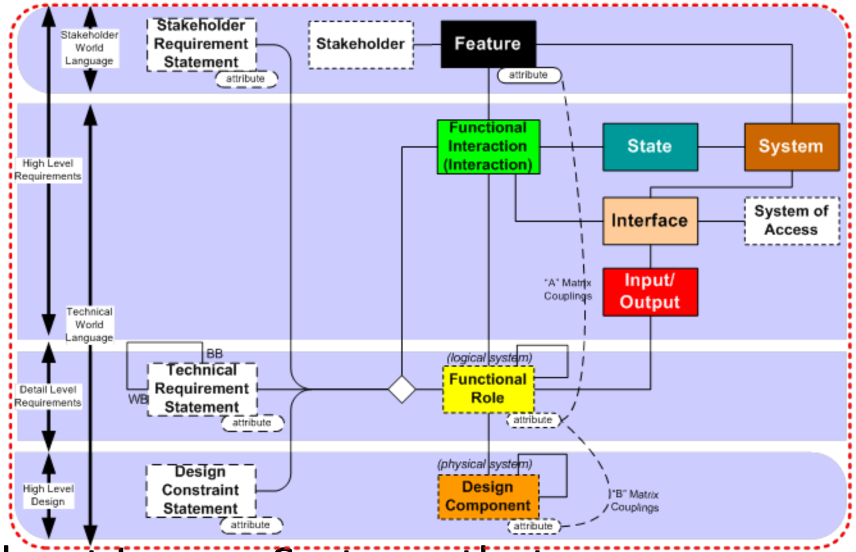
What are S*Patterns?

- Connecting our two Working Groups:
  – PLE information can be captured and expressed as configurable S*Patterns.
  – This includes their capture and distillation from Legacy Systems



What do S*Patterns bring to MBSE representations of Legacy Systems?

- Recurring patterns in Legacy Systems are about more than architectural patterns alone.
- Among the things they allow us to represent in MBSE S*Pattern form are:
  - Fitness space (Features)
  - Interactions
  - Interfaces
  - States (modes, phases)
  - Requirements
  - Logical and Physical Architecture
  - Attribute (parametric) Couplings
  - Simulatable behavior
  - Selection and Evolution



- Gestalt Rules for Patterns
- Including model capture of observations about Legacy Systems that may otherwise exist only in minds or transient artifacts.

Examples—past and future

- As reported in IS2011, projection of legacy systems onto extracted S*Patterns simultaneously adds to spec completeness (***expanding***) while also ***compressing*** legacy information!

Question: How can that be so?

What Is the Smallest Model of a System?

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

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

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

Introduction and Background: Size Matters!

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

Language and Compression. This subject may appear to be related to the language used to describe systems, and an interesting thread in the mathematical study of description length is whether minimality is in a sense independent of language (Chaitin, Grunwald, Li and Vitany). In any case, systems modeling languages such as SysML® and its predecessors provide

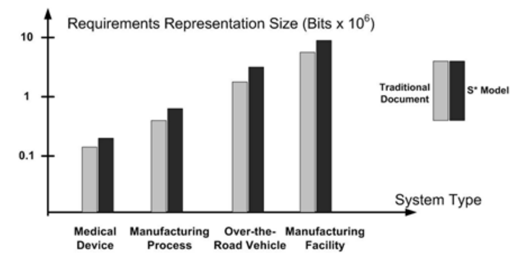


Figure 9: Typical sizes for models and traditional systems engineering documents

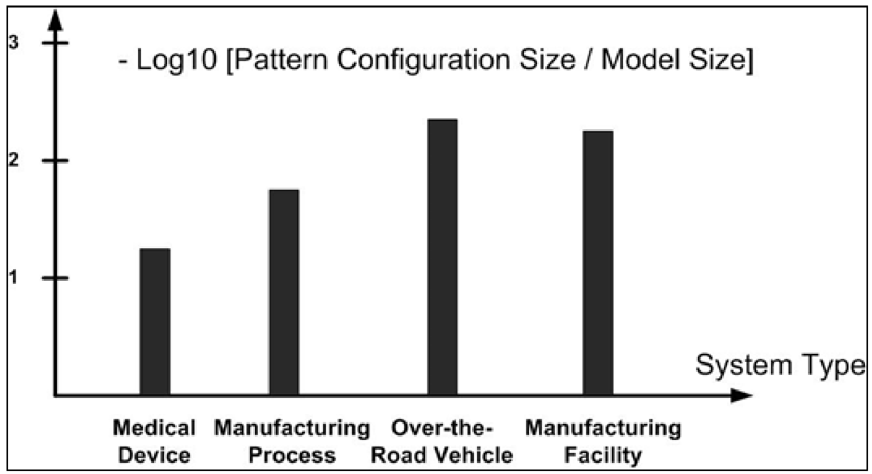
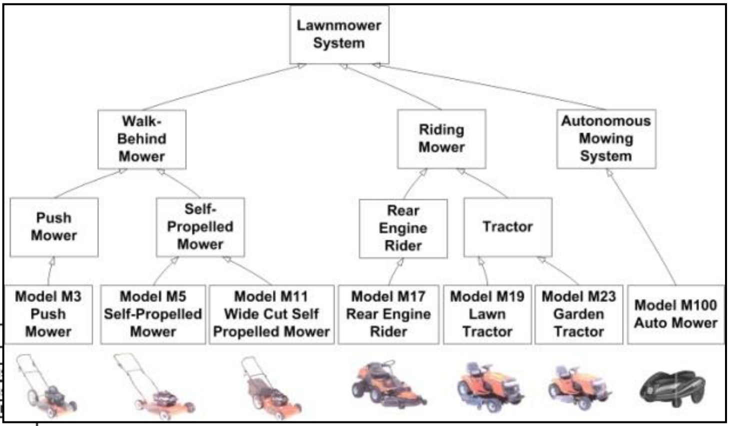


Figure 12: Pattern Compression

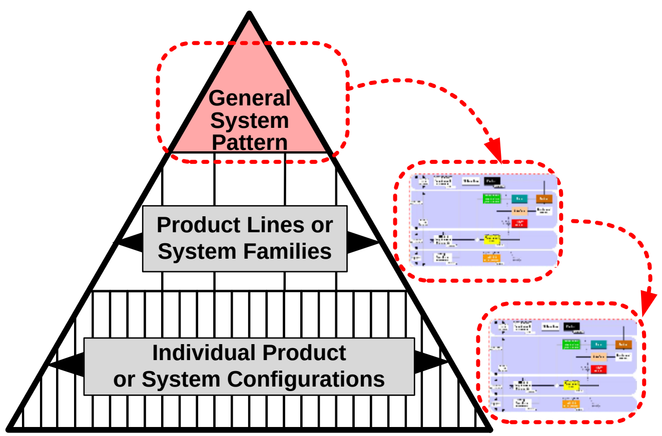
Answer: PBSE splits information into underlying pattern (fixed) and configuration (variable) data:

- The variable pattern data then gets the most attention
- The variable pattern data is much smaller than the fixed data
- The fixed underlying pattern data is still there for reference



Lawnmower Product Line: Configurations Table

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



System Pattern Class Hierarchy

A public example

- The above example statistics are drawn from S*Patterns confidential to individual enterprises.
- The Patterns Working Group is interested in constructing a public example of an MBSE PLE S*Pattern that is extracted from legacy systems:
  - A natural project to collaborate on with the PLE Working Group, if interested . . .

Discussion of ideas and interests by the PLE WG and the Patterns WG membership, in potential joint projects

Idea 1: Joint construction and analysis of a public example legacy PLE system pattern.

Other Ideas:

-
-
-

Product Line Engineering References:

1. ISO/IEC 26550:2013, "Software and systems engineering -- Reference model for product line engineering and management", 2013.
2. INCOSE PLE Working Group charter: http://www.incose.org/docs/default-source/wgcharters/product-lines.pdf?sfvrsn=6
3. Hetrick, W., Krueger, C., and Moore, J.,"Incremental Return on Incremental Investment: Engenio's Transition to Software Product Line Practice", ACM OOPSLA'06 October 22–26, 2006, Portland, Oregon, USA.

S*Patterns and PBSE References:

1. INCOSE Patterns Working Group web site: http://www.omgwiki.org/MBSE/doku.php?id=mbse:patterns:patterns
2. INCOSE Patterns Working Group, "MBSE Methodology Summary: Pattern-Based Systems Engineering (PBSE), Based On S*MBSE Models", V1.5.5A, retrieved 2016 from: http://www.omgwiki.org/MBSE/doku.php?id=mbse:pbse
3. Schindel, W., "What Is the Smallest Model of a System?", *Proc. of the INCOSE 2011 International Symposium*, 2011.
4. Schindel, W., "Requirements statements are transfer functions: An insight from model-based systems engineering", *Proceedings of INCOSE 2005 International Symposium*, 2005.
5. Peterson, T., and Schindel, W., "Introduction to Pattern-Based Systems Engineering (PBSE): Leveraging MBSE Techniques", in *Proc. of INCOSE 2013 International Symposium*, Tutorial, June, 2013.