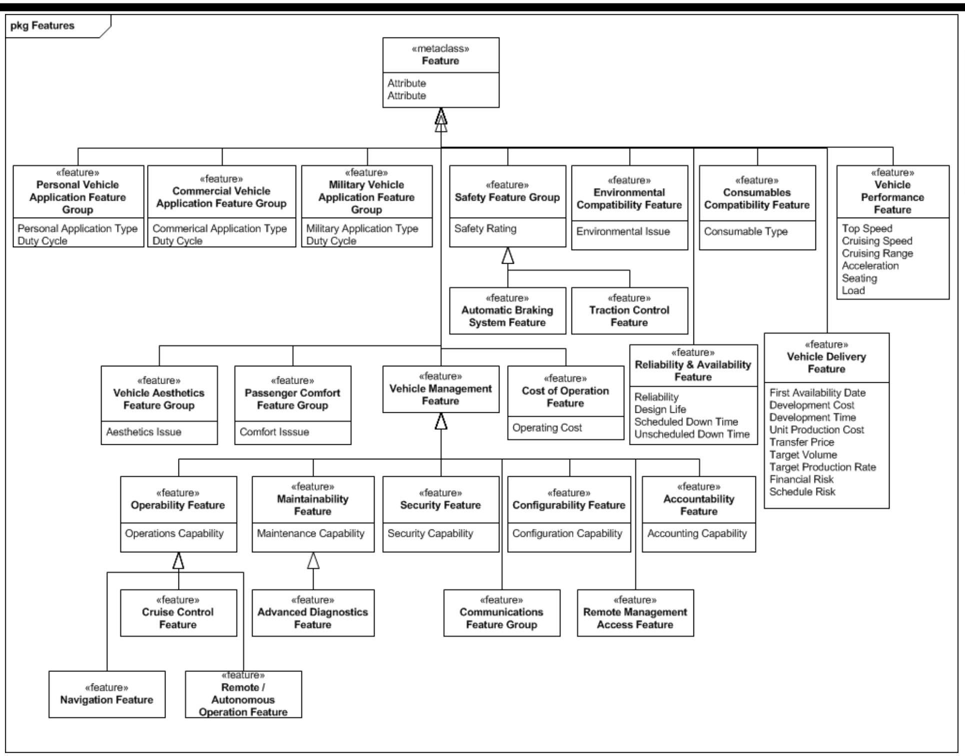
ICTT System Sciences
Bill Schindel
schindel@ictt.com

Booz | Allen | Hamilton
strategy and technology consultants
Troy Peterson
peterson_troy @ bah.com

# Pattern-Based Systems Engineering (PBSE)
## speed - leverage - knowledge





Cleveland Chapter Event 5/19/15

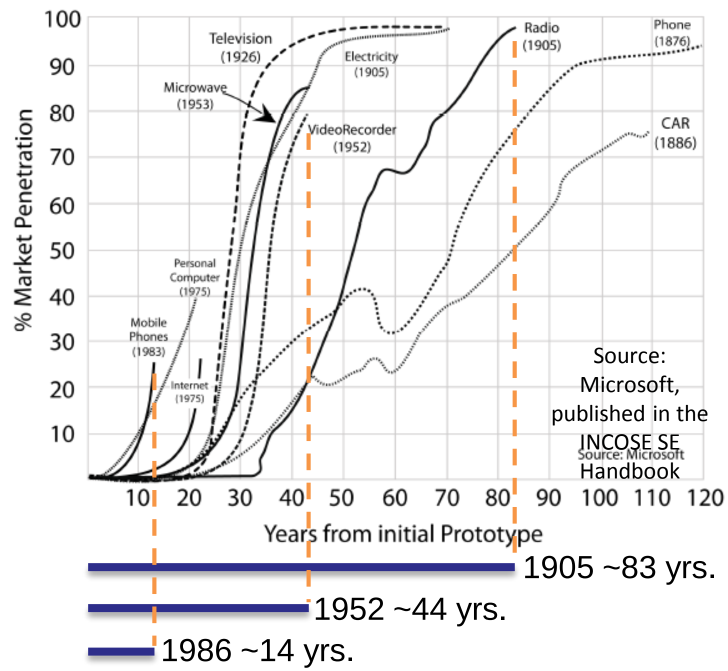
# Contents

# PBSE: The Need, Call-to-Arms, and Vision

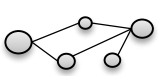
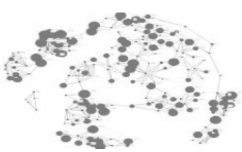
- INCOSE thought leaders have discussed the growing need to address 10:1 more complex systems with 1:10 reduction in effort, using people from a 10:1 larger community than the “systems expert” group

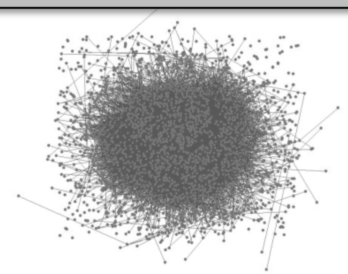
- Many SE efforts are in some way concerned with growing complexity, but none give evidence of the sweeping order-of-magnitude improvements demanded by this call-to-arms.

- PBSE is a methodical way to achieve this order-of-magnitude improvement



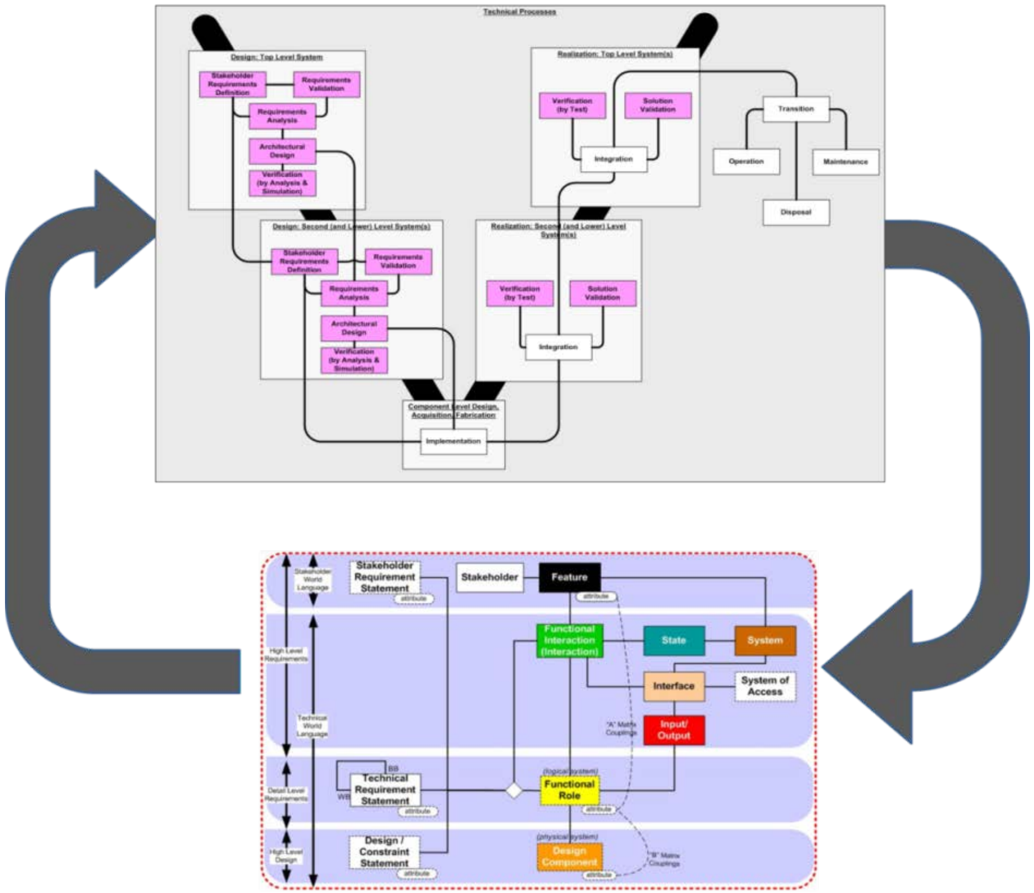

***Time to market penetration decreased by 4x over 50 years***



# A shift in emphasis is underway

Innovation Process




Information Passing Through the Innovation Process

- Shifting from historical emphasis on the SE Process . . . .
- To emphasis on Model Information flowing through that process.

# PBSE Enablers: Model-Based Systems Engineering (MBSE)

- MBSE as “the <u>*formalized application of modeling*</u> to support system requirements, design, analysis, verification and validation activities

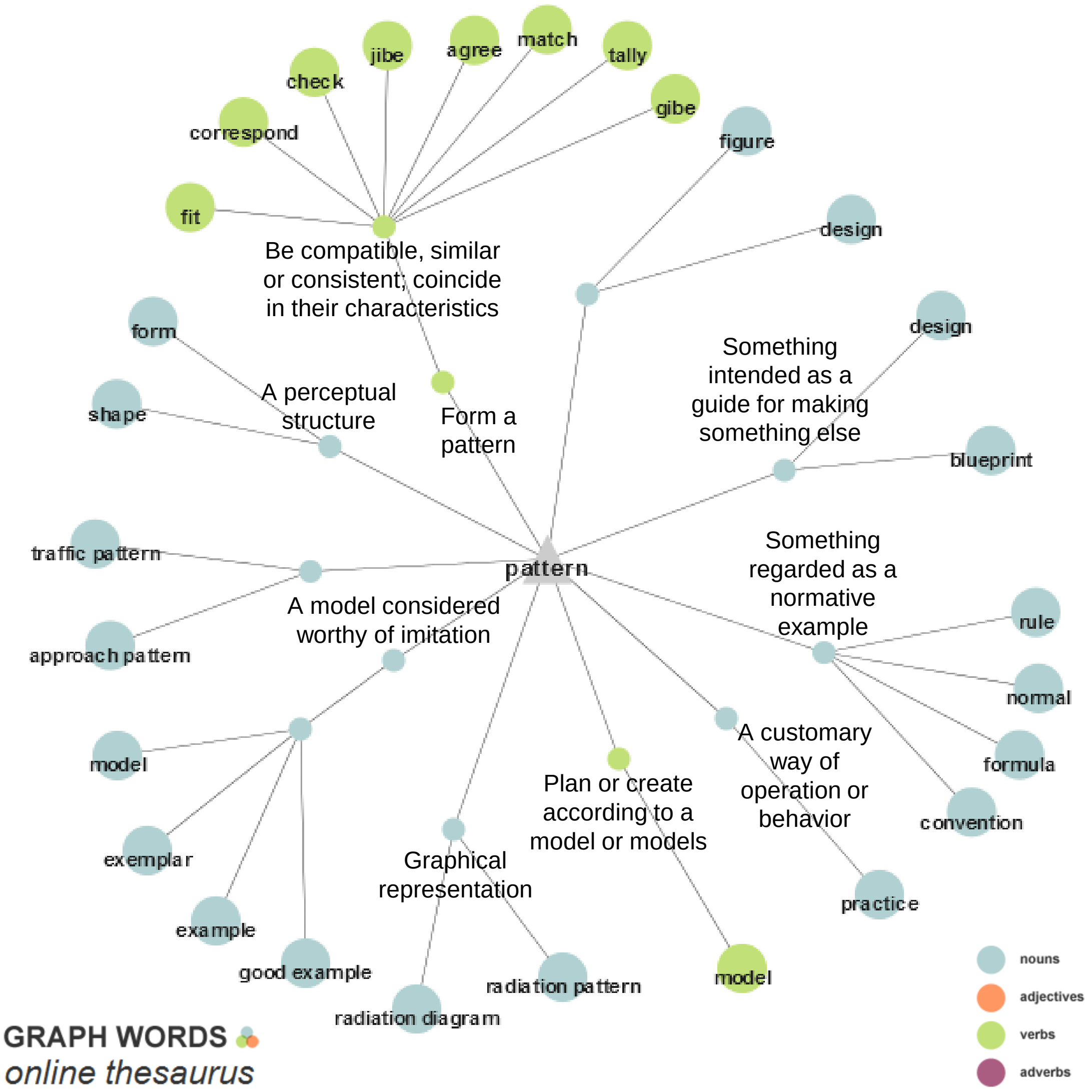
- MBSE is part of a long-term trend toward <u>*model-centric approaches*</u> adopted by other engineering disciplines, including mechanical, electrical and software.

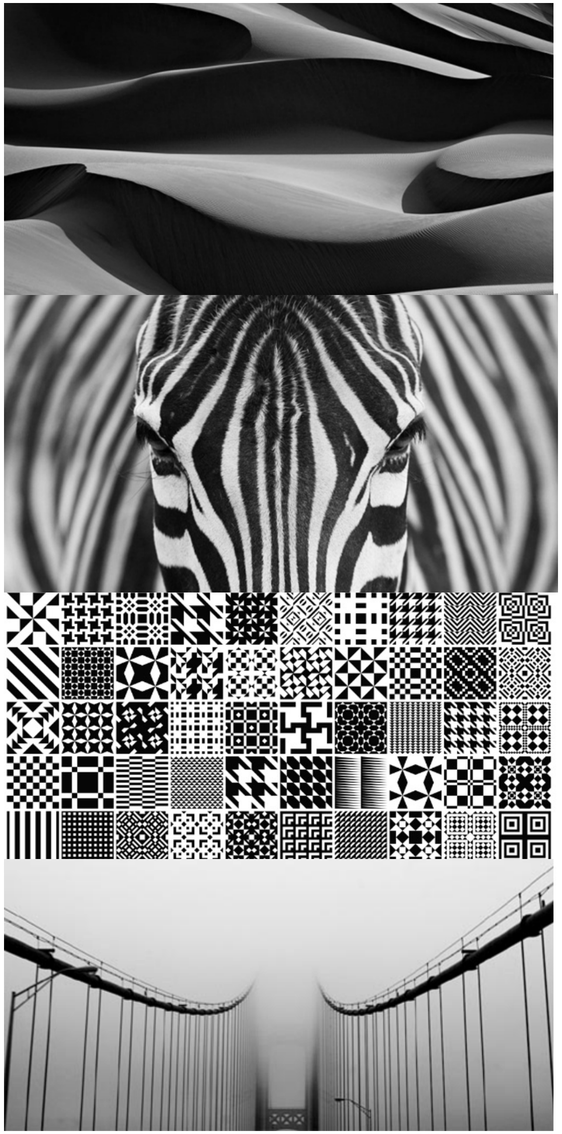
- MBSE is expected to replace the document-centric approach and to <u>*influence the future practice*</u> of systems engineering

- MBSE is expected to <u>*provide significant benefits*</u> over the document centric approach by enhancing productivity and quality, reducing risk, and providing improved communications among the system development team.

http://www.omgwiki.org/MBSE/doku.php

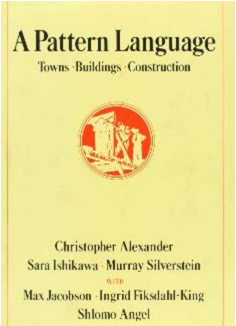
# Patterns...

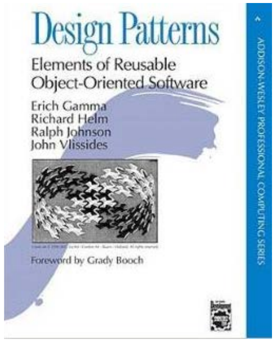

pattern
Be compatible, similar or consistent; coincide in their characteristics
fit
correspond
check
jibe
agree
match
tally
gibe
figure
design
Something intended as a guide for making something else
design
blueprint
Form a pattern
A perceptual structure
form
shape
A model considered worthy of imitation
traffic pattern
approach pattern
model
exemplar
example
good example
Graphical representation
radiation diagram
radiation pattern
Plan or create according to a model or models
model
Something regarded as a normative example
rule
normal
formula
convention
A customary way of operation or behavior
practice
nouns
adjectives
verbs
adverbs
GRAPH WORDS
online thesaurus

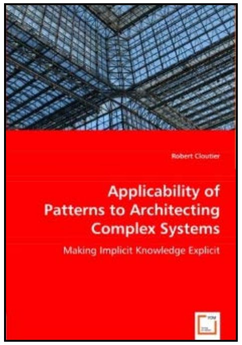


# What might “pattern” mean for systems?

- 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 and time.
- Various forms of representation.
- However, when we refer to “PBSE” in this presentation, we will  mean the use of  <u>S*Patterns</u> . . . .

# The systems engineering connection

- *Discovering regularities and how to represent them* has been at the heart of science and engineering progress:
  - During 2012-13, the INCOSE System Sciences Working Group (SSWG) bridged related interests of engineering and science.
  - The INCOSE Patterns Challenge Team of the MBSE Initiative formed in 2013, performed Wave 1 projects in 2014, starting Wave 2 projects in 2015.

- *Ability to manage risk and adapt* are related to our *awareness* and *understanding* of the *regularities* (patterns) around us:
  - Whether in the systems we engineer, or the markets and operational environments in which their life cycle unfolds.
  - They exert "forces" on us, whether are aware of them or not.

# What repeating regularities are of interest?

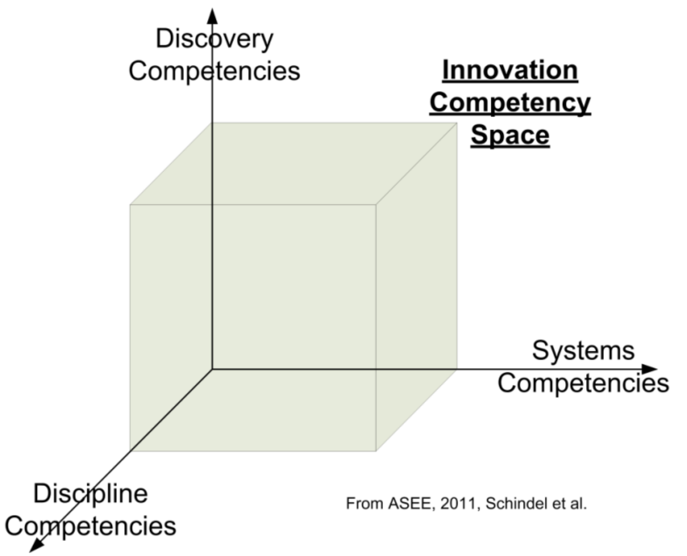
- Smaller-Scale Regularities:
  - Patterns of Stakeholder Features (e.g., in vehicles, energy systems, etc.)
  - Patterns of Requirements
  - Patterns of Design Solutions
  - Patterns of Failure Modes and Effects
  - Patterns of Functional Roles, Interactions, States
  - Patterns of Interfaces, Input-Outputs, and Access
  - Patterns of Technologies
- Larger-Scale Regularities:
  - Patterns of how all the above are related to each other
  - Patterns in couplings across systems, domains, SOS's
  - Systems of Material Handling, Production, Distribution, Sustainment
  - Systems of Innovation
  - Patterns of Systems Pathologies

# Is this "just of academic interest"?

- Hardly! Lack of awareness of these regular patterns leaves products, programs, enterprises at serious risk:
  - Re-experiencing the same mis-steps and reworks;
  - Just because we have made one system work, how do we know what will happen when we deploy more of them, as markets, conditions, & technologies evolve?
  - Just because our system has human experts on hand today, how do we know what will happen when they move on?
- Example cases and responses:
  - FDA push to the pharmaceutical manufacturing industry to improve the science-based understanding of underlying process transformations, provable ranges, and control strategies, etc.
  - The generation of system requirements families for globally-deployed product families and their production, distribution, and support systems.
  - The generation of system verification plans from underlying patterns of system requirements.
  - The use of System Patterns to generate Risk Analyses (e.g., FMEAs, etc.) for a variety of domain systems.

# “Chance favors the prepared mind” - Louis Pasteur

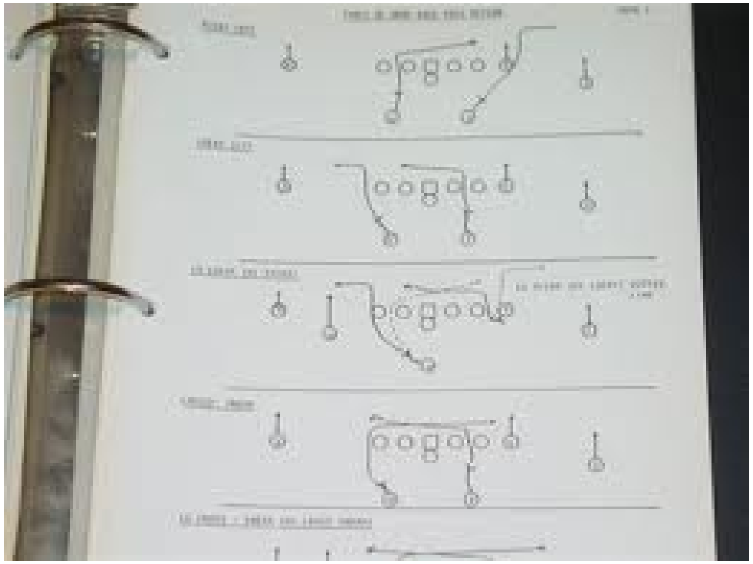
- Explicit patterns help us organize what we know--as well as what we don’t.
- Explicit preparation for:
  - System & program risks
  - Market & competitive shifts
  - New science & technology
  - Life cycle extensions
- Adaptability!




From ASEE, 2011, Schindel et al.

# Adaptation Response Time

- Explicit pattern awareness helps us to:
  - Recognize the situation has changed.
  - Know the best alternate pattern configuration.

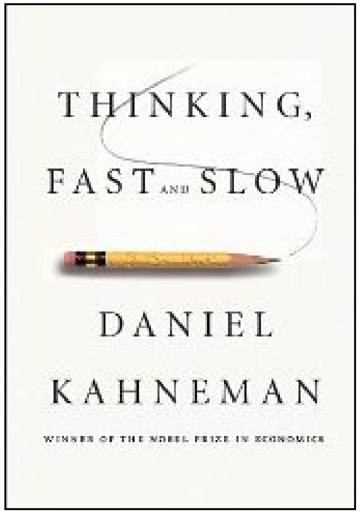


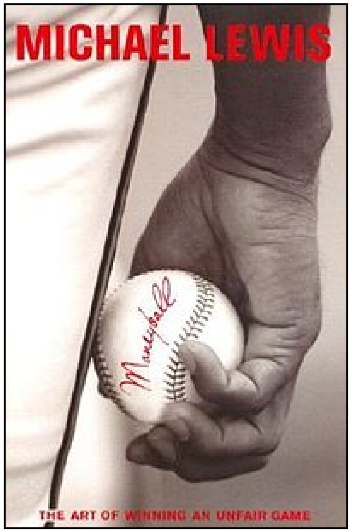
# Irrationality: Human beings' behaviorally-preferred mode?

- A broad issue across human life:
  - The science of irrationality
  - Daniel Kahneman, Nobel Laureate, "Thinking, Fast and Slow")
  - "Moneyball", Oakland A's, Billy Beane.

- Engineering teams more rational than others?
  - Ever encounter a bad decision?
  - A significant fraction of requirements are left unstated

- Patterns existing in Nature do not mean the patterns are recognized by humans



# One way people cope . . .

- "Domain experts" internalize patterns:
  - These human experts influence our projects, using their experience, intuition, informed judgment.


# System Patterns: Dark and Visible

- The regularities are “out there”, whether we represent them or not:
  - In particular, they impact our ability to deal with uncertainty and adaptability.
- We use the term Dark Pattern to refer to system regularities that have not been explicitly represented:
  - They are in a sense “invisible”, but still impact our systems, customers, programs, enterprises, institutions, and society.
- By contrast, when we represent those System Patterns formally, they become “visible”, as Explicit Patterns:
  - Our method for doing this is Pattern-Based Systems Engineering (PBSE);
  - PBSE is an extension of Model-Based Systems Engineering (MBSE);
  - PBSE creates and applies configurable, re-usable models, called Patterns;
  - They typically include much more than just the “subject system”.

# How many patterns are Dark?

- Most systems programs involve Patterns, such as:
  - Patterns of available technologies and parts
  - Patterns of candidate solution architectures
  - Patterns of interfaces
  - Patterns of system states or modes
  - Patterns of customers, or market expectations
  - Patterns of competitive offerings
  - Patterns of system failures modes and effects
- Most systems engineering efforts—even model-based--still occur without use of explicit Pattern-Based methods:
  - This is the world of Dark Patterns.
  - Example: Nearly universally missed requirements.
- Explicit Patterns prepare us to adapt by describing key objects, relationships, and variables—including multiple types of risk.

# Representing System Patterns

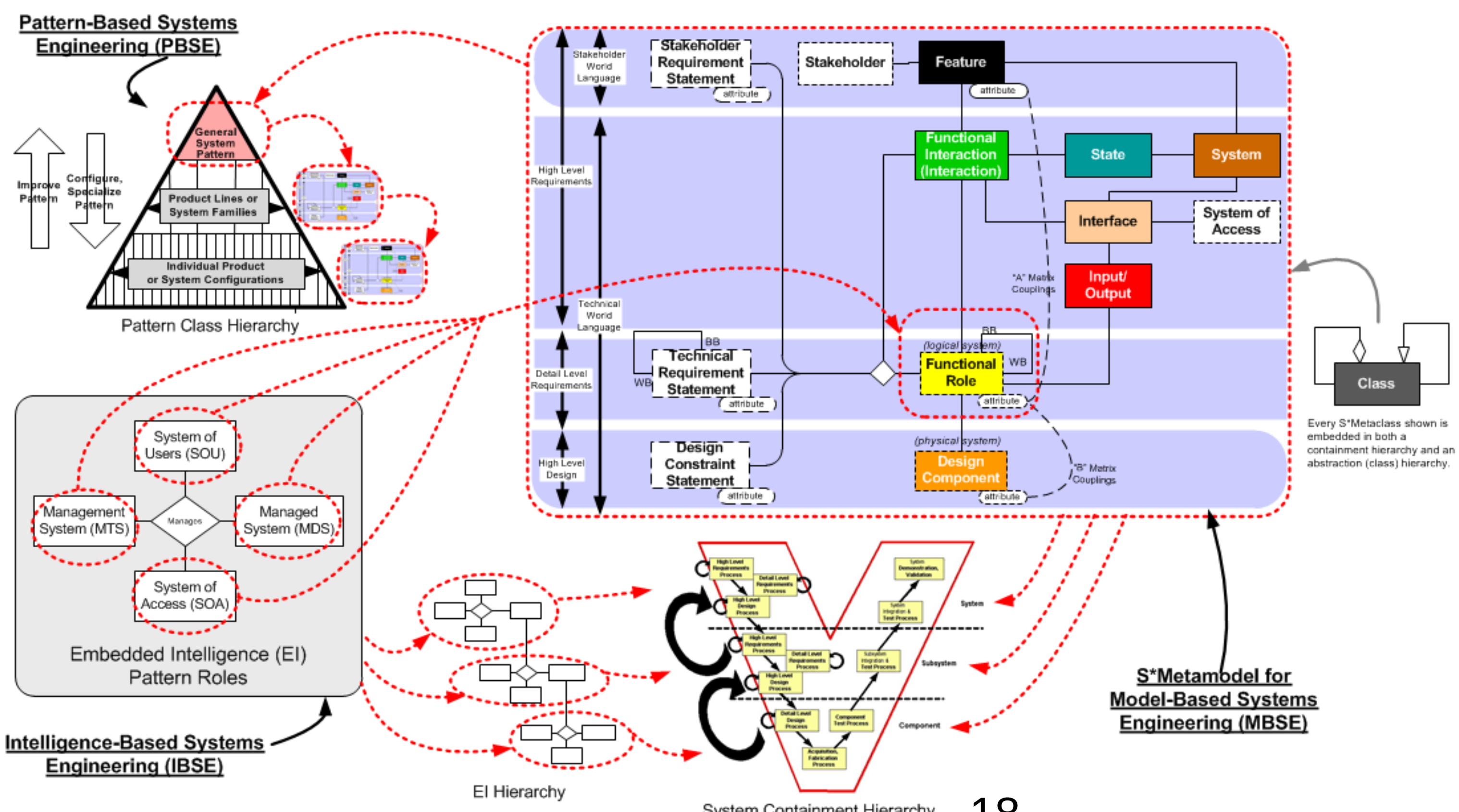
- What is the smallest amount of information we need to represent these regularities?
  - Some people have used prose to describe system regularities.
  - This is better than nothing, but usually not enough to deal with complex systems.

- We use S* Models, which are the minimum model-based information necessary:
  - This is not a matter of modeling language—your current favorite language and tools can readily be used for S* Models.
  - The minimum underlying information classes are summarized in the S* Metamodel, for use in any modeling language.

- The resulting system model is made configurable and reusable, thereby becoming an S* Pattern.

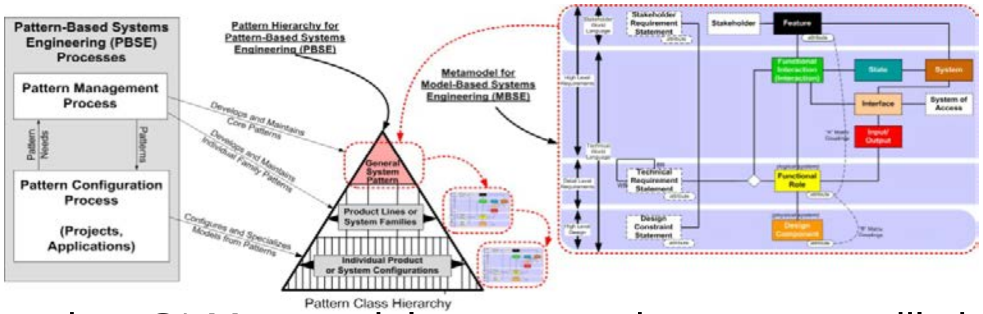
# Pattern-Based Systems Engineering (PBSE)

- S*Patterns are formally configurable, using automated algorithms, portable across numerous third-party COTS engineering tools and databases, to rapidly generate many specific system requirement/design configurations (including failure mode analyses) from desired platform features:



# Concept Summary: PBSE

- The PBSE **approach respects the systems engineering tradition**, body of knowledge, and historical lessons, while providing a high-gain path forward**.**
- An S* Pattern is a configurable, re-usable S* Model (S*Metamodel compliant). It is an **extension of the idea of a Platform** (which is a configurable, re-usable design). The Pattern includes not only the Platform, but all the extended system information (e.g., requirements, risk analysis, design trade-offs & alternatives, decision processes, etc.):




- By including the appropriate S* Metamodel concepts, these can readily be managed in (SysML or other) preferred modeling languages and tools—the **ideas involved here are not specific to a modeling language or specific tool.**
- The order-of-magnitude changes have been realized because projects that use PBSE rapidly **start from an existing Pattern, gaining the advantages of its content,** and feed the pattern with what they learn, for future users.
- The "game changer" here is the shift from **"learning to model" to "learning the model",** freeing many people to rapidly configure, specialize, and apply patterns to deliver value in their model-based projects.

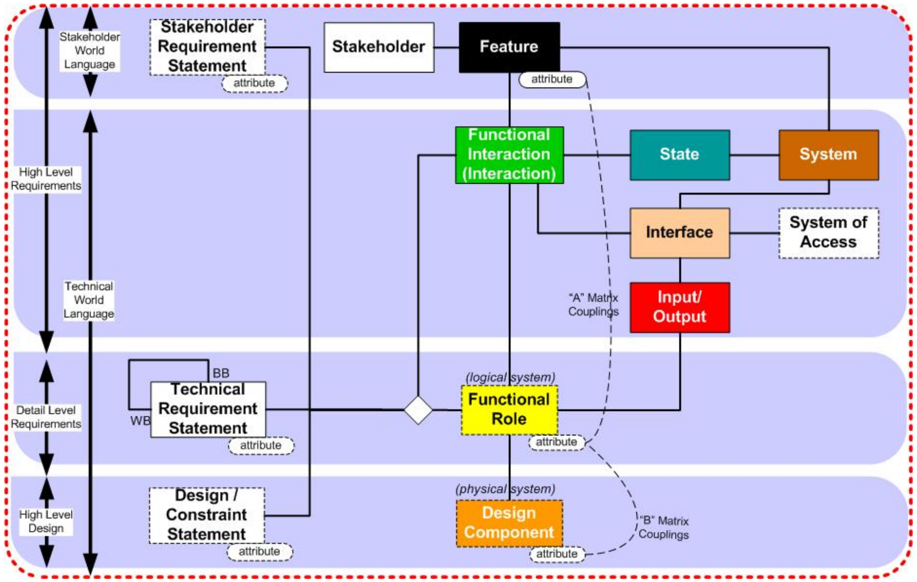
# Constructing an efficient representation

- A metamodel is a model of other models;
  - Sets forth how we will represent Requirements, Designs, Verification, Failure Analysis, Trade-offs, etc.;
  - We utilize the (language independent) S* Metamodel from Systematica™ Methodology:

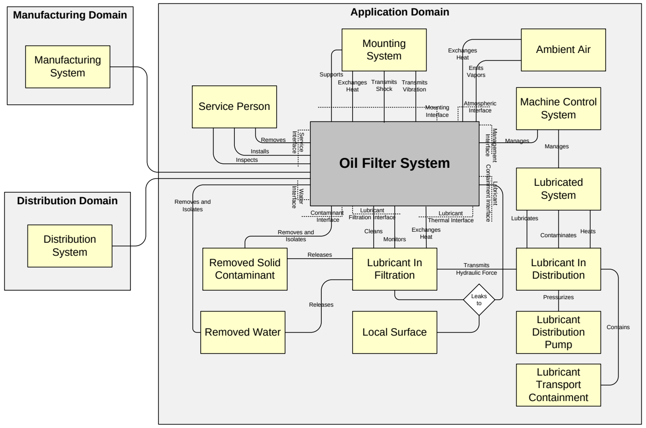
- The resulting system models may be expressed in SysML™, other languages, DB tables, etc.

- Has been applied to systems engineering in aerospace, transportation, medical, advanced manufacturing, communication, construction, other domains.

Simple summary of detailed S* Metamodel.



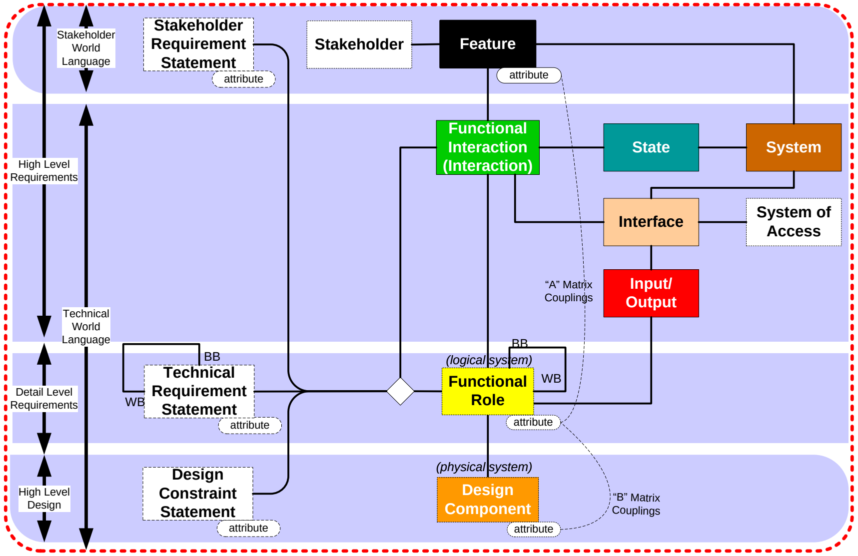
- S*Patterns are Model-Based:
  - We are referring to patterns represented by formal system models.




  - Many of the historical "design patterns" were not based on formal system models, but prose, in simple templates.
  - S*Patterns are model-based, but not dependent on any single system modeling language, and are readily expressed in SysML, IDEF, or other formal modeling languages.

# Pattern-Based Systems Engineering (PBSE)

- S*Patterns are <u>Model-Based</u>:
  - Independent of the specific modeling language, S*Models always conform to the underlying S*Metamodel:

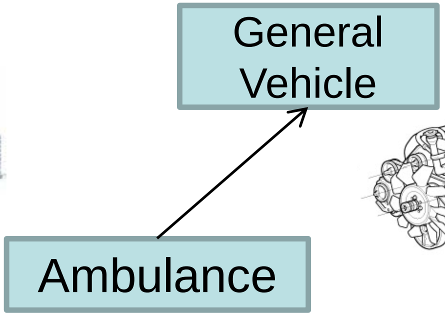


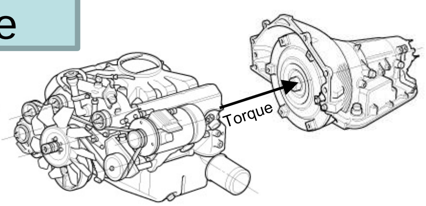

Summary view of S*Metamodel

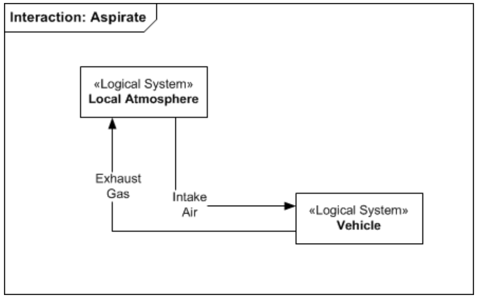
  - The S*Metamodel is the smallest model sufficient to the purposes of engineering and science.

# PBSE Enablers: Definitions of some S* Metamodel Classes

- **<u>System</u>**: A collection of interacting components. Example: Vehicle; Vehicle Domain System.
- **<u>Stakeholder</u>**: A person or other entity with something at stake in the life cycle of a system. Example: Vehicle Operator; Vehicle Owner; Pedestrian
- **<u>Feature</u>**: A behavior of a system that carries stakeholder value. Example: Automatic Braking System Feature;  Passenger Comfort Feature Group
- **<u>Functional Interaction (Interaction):</u>** An exchange of energy, force, mass, or information by two entities, in which one changes the state of the other. Example: Refuel Vehicle;  Travel Over Terrain
- **<u>Functional Role (Role):</u>** The behavior performed by one of the interacting entities during an Interaction.  Example:  Vehicle Operator; Vehicle Passenger Environment Subsystem
- **<u>Input-Output:</u>** That which is exchanged during an interaction (generally associated with energy, force, mass, or information). Example: Fuel, Propulsion Force, Exhaust Gas

# PBSE Enablers: Definitions of some S* Metamodel Classes

- **System of Access:** A system which provides the means for physical interaction between two interacting entities. Examples: Fueling Nozzle-Receptacle; Grease Gun Fitting; Steering Wheel; Dashboard; Brake Peddle

- **Interface:** The association of a System (which "has" the interface), one or more Interactions (which describe behavior at the interface), the Input-Outputs (which pass through the interface), and a System of Access (which provides the means of the interaction). Examples: Operator Interface; GPS Interface

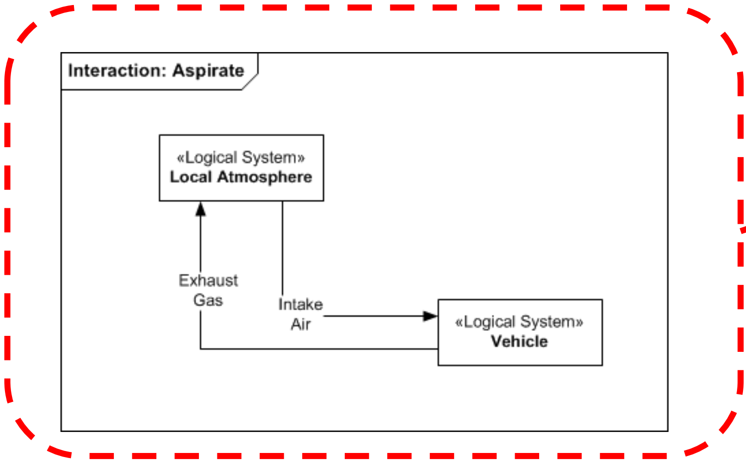
- **State:** A mode, situation, or condition that describes a System's condition at some moment or period of time. Example:  Starting; Cruising; Performing Maneuvers

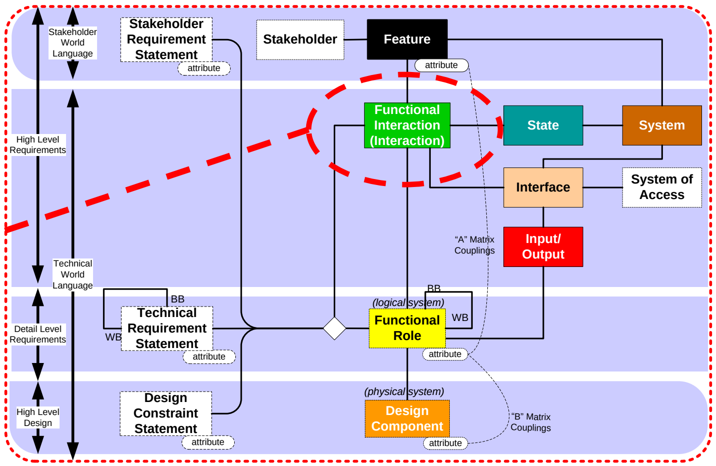
- **Design Component:** A physical entity that has identity, whose behavior is described by Functional Role(s) allocated to it. Examples: Garmin Model 332 GPS Receiver; Michelin Model 155 Tire

- **Requirement Statement:** A (usually prose) description of the behavior expected of (at least part of) a Functional Role. Example: "The System will accept inflow of fuel at up to 10 gallons per minute without overflow or spillage."

# Pattern-Based Systems Engineering (PBSE)

- The S*Metamodel explicates Physical Interactions:
  - Interactions: state-impacting exchange of energy, force, mass, or information:



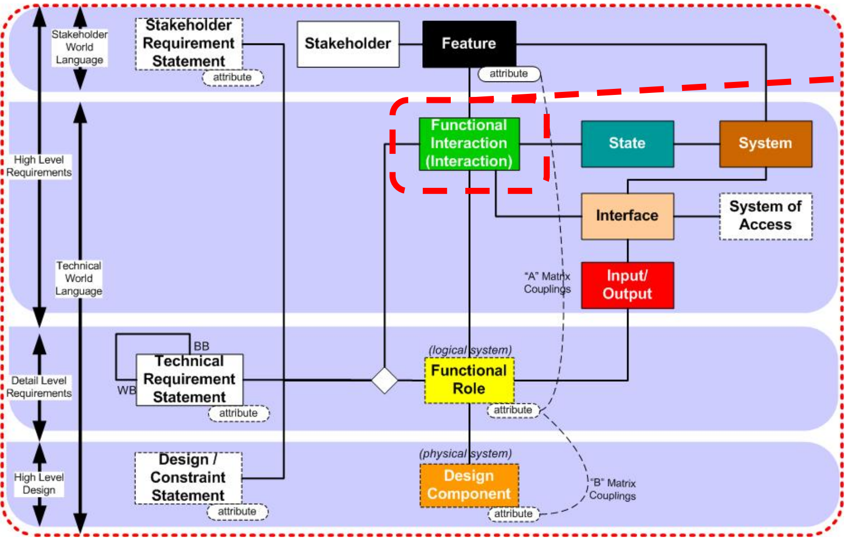



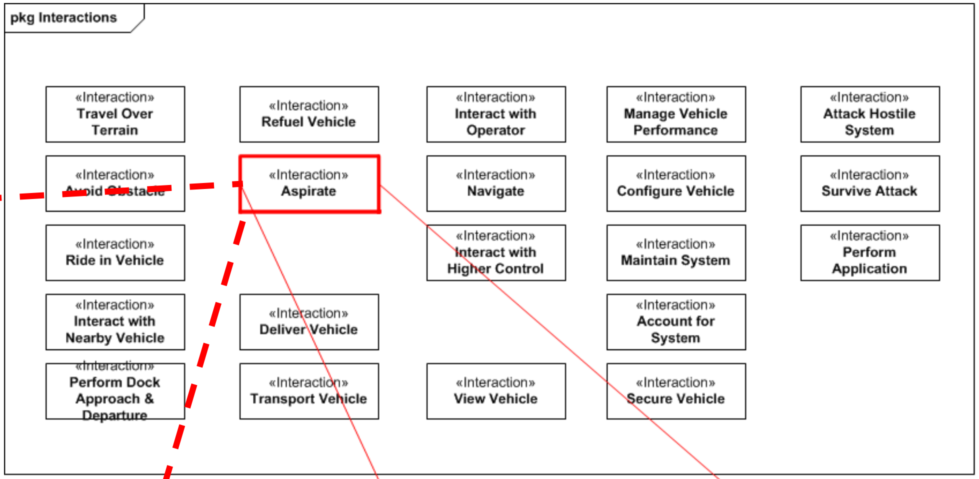

Summary view of S*Metamodel

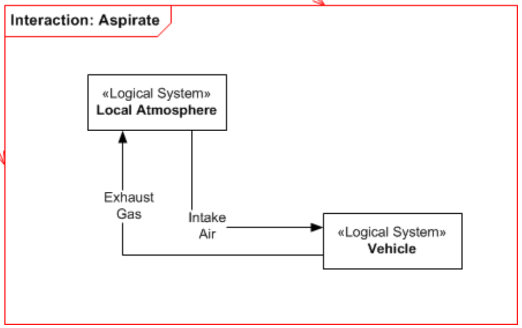
  - Such interactions are the basis of substantially all the laws (patterns, regularities) of the physical sciences.
  - Systems Engineering should have as strong a foundation as the other engineering disciplines.

# Physical Interactions: At the heart of S* models

- S* models represent Physical Interactions as explicit objects:



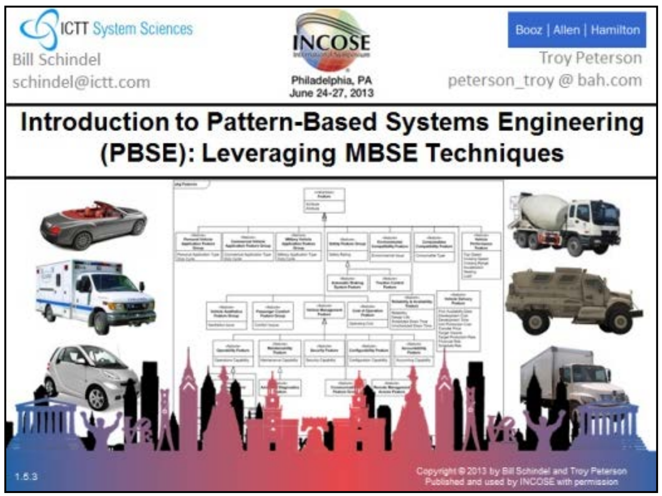



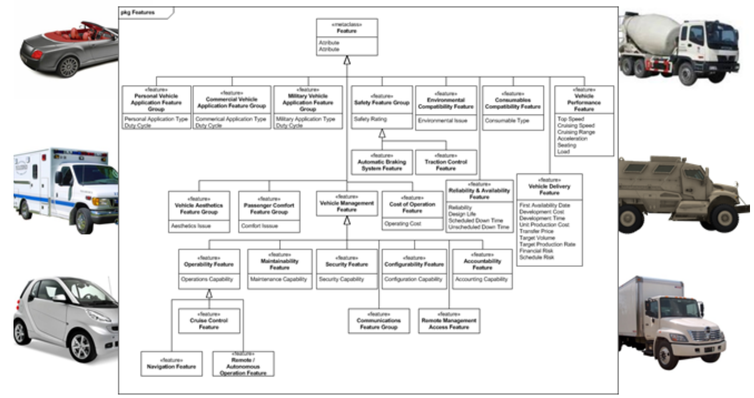

**Aspirate**: The interaction of the vehicle with the Local Atmosphere, through which air is taken into the vehicle for operational purposes, and gaseous emissions are expelled into the atmosphere.

# Pattern-Based Systems Engineering (PBSE)

- The <u>scope</u> of S*Patterns are "<u>Whole Systems</u>":
  - An S*Pattern is effectively a formal model of a <u>platform</u> system, or a whole system domain:



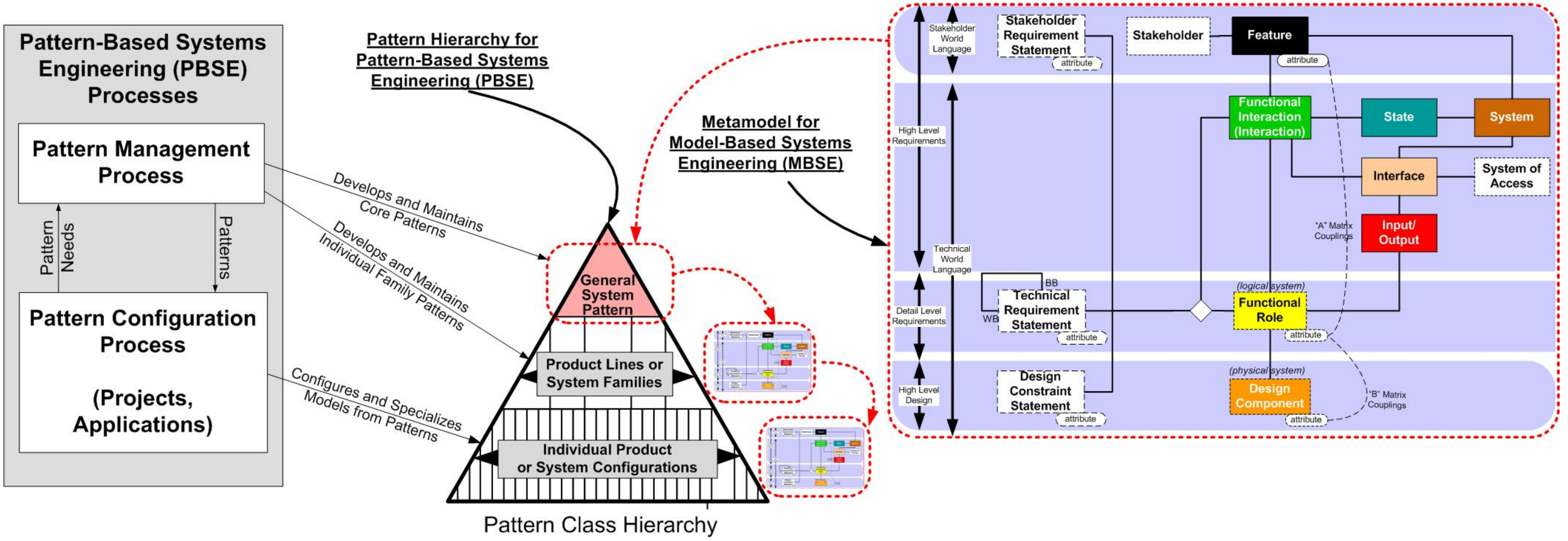

A vehicle pattern in SysML



  - Historical "design patterns" were most frequently about smaller repeating component or subsystem patterns, used as deemed applicable.
- The <u>scope</u> of S*Patterns includes system requirements, designs, and other S*Model information such as verification, failure analysis, etc.

# Pattern-based systems engineering (PBSE)

- Pattern-Based Systems Engineering (PBSE) has two overall processes:
  - **Pattern Management Process**: Generates the underlying family model, and periodically updates it based on application project discovery and learning;
  - **Pattern Configuration Process**: Configures the pattern into a specific model for application in a project.

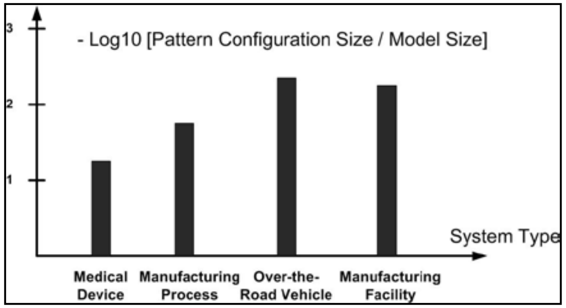


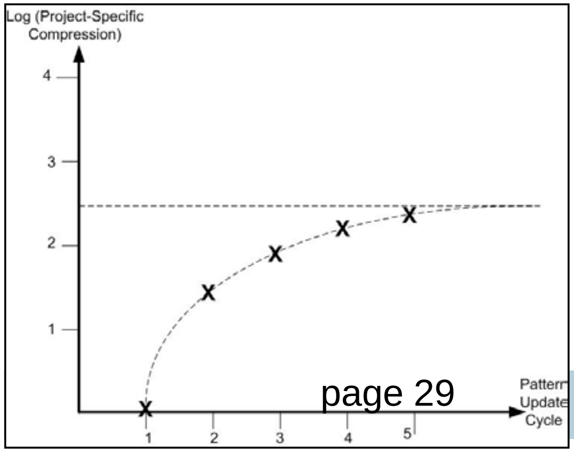
# Pattern configurations

- A table of configurations illustrates how patterns facilitate compression;
- Each column in the table is a compressed system representation with respect to ("modulo") the pattern;
- The compression is typically very large;
- The compression ratio tells us how much of the pattern is variable and how much fixed, across the family of potential configurations.

**Lawnmower Product Line: Configurations Table**

| | | Units | Walk-Behind Push Mower Push Mower | Walk-Behind Mower Self-Propelled | Walk-Behind Self-Propelled Wide Cut | Riding Rider Rider | Riding Tractor Lawn | Riding Mower Tractor Garden | Autonomous Autonomous 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 | [illegible] | 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 | |



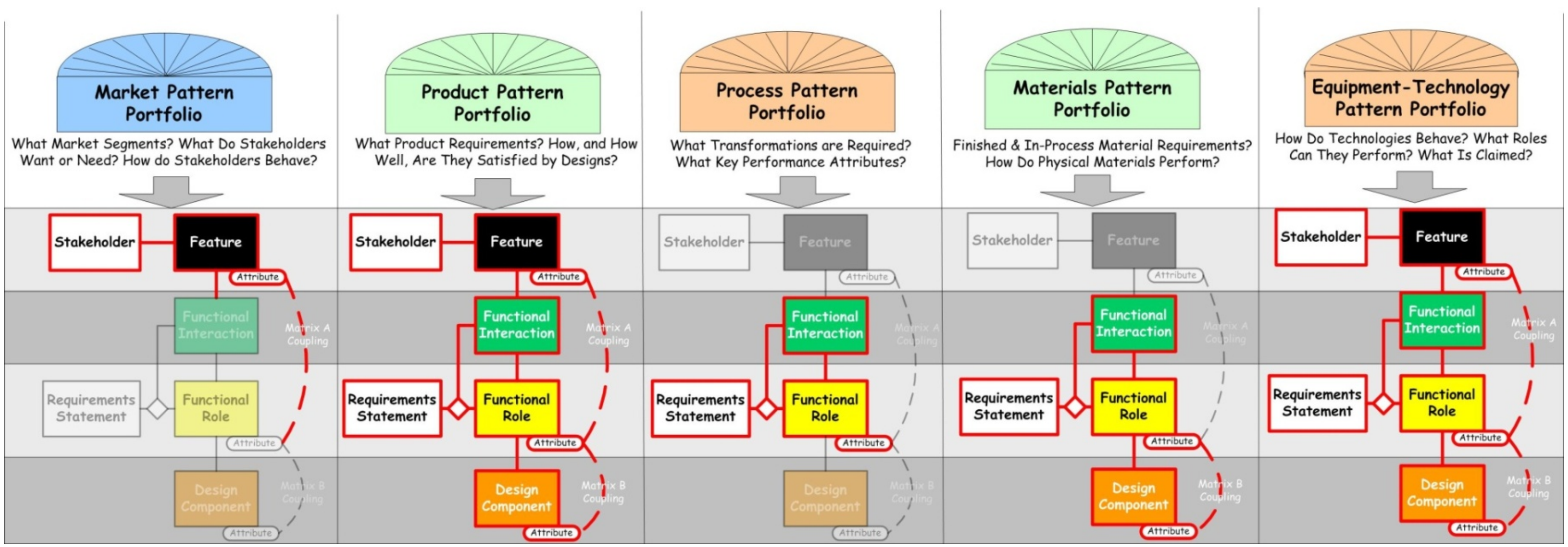
# Benefits of applying system patterns

- Example Uses and Benefits:
  1. Stakeholder Features and Scenarios: Better stakeholder alignment sooner
  2. Pattern Configuration: Generating better requirements faster
  3. Selecting Solutions: More informed trade-offs
  4. Design for Change: Analyzing and improving platform resiliency
  5. Risk Analysis: Pattern-enabled FMEAs
  6. Verification: Generating better tests faster

- Practice PBSE with a goal in mind: What benefits seem most important?

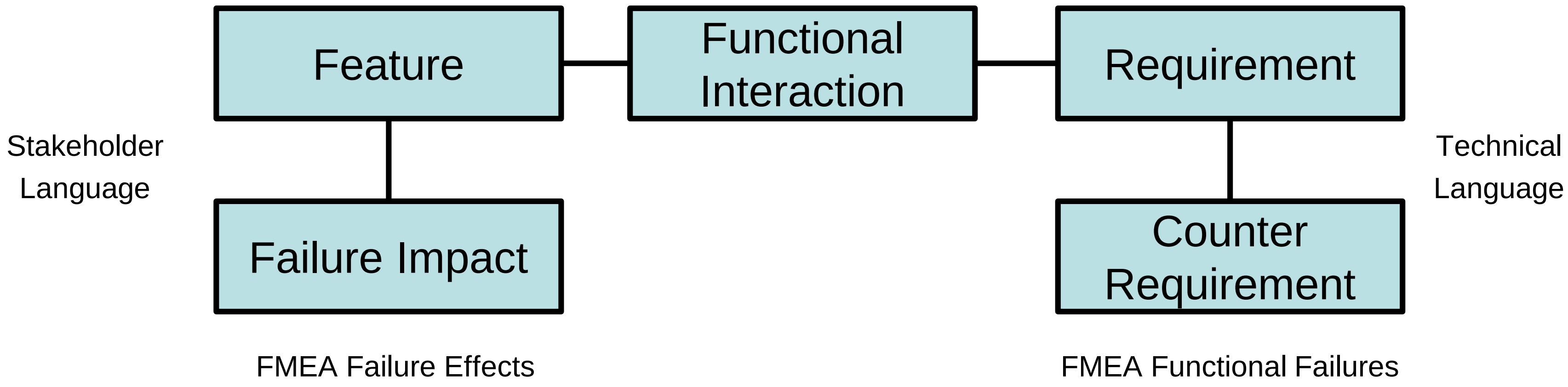
# Patterns organize portfolios

- Patterns express “envelopes” around “point situations”.
- Patterns help us discover, explore, and record what we may have to adapt to, along with adaptation plans:
  - Evolution in available technologies and parts
  - Evolution in system requirements, interfaces, modes, etc.
  - Evolution in the larger systems in which we operate
  - Evolution in customer or market expectations
  - Evolution in competitor offerings



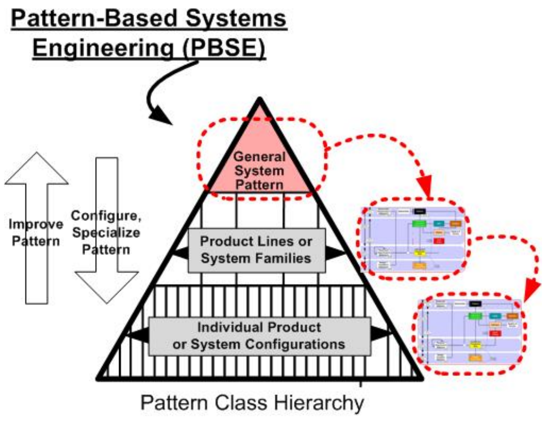
# Patterns for managing risk

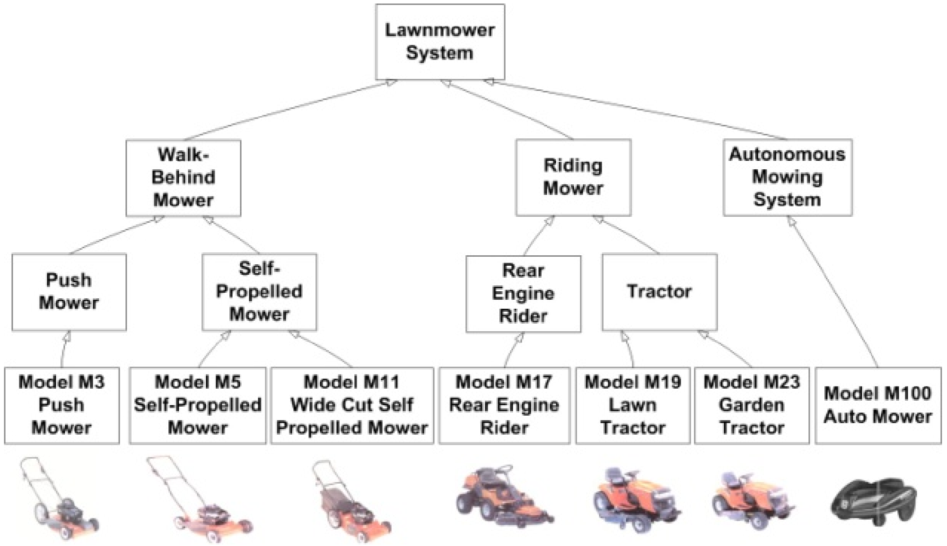
- Patterns also express risks and mitigations for:
  - Patterns of system failure modes and effects (d-FMEA)
  - Patterns of operator failure modes and effects (a-FMEA)
  - Patterns of production & distribution failures (p-FMEA)



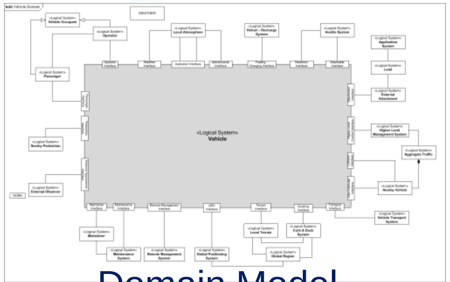
# PBSE helps make Platform Management a discipline

- Descriptions of SE processes typically appear to describe engineering a "new" system "from scratch" [e.g., ISO 15288, INCOSE SE Handbook]:
  - However, real projects are often concerned with engineering similar (but different) systems across different product generations, applications, configurations, or market segments.
  - Patterns provide the IP basis to make Platform Management a discipline, not just an attractive idea:

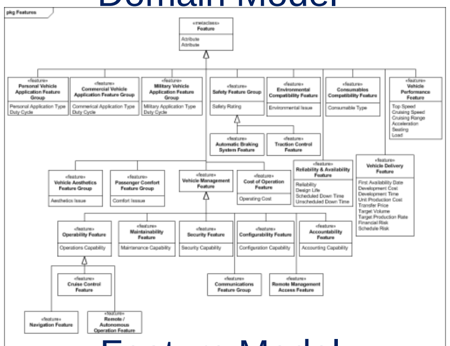


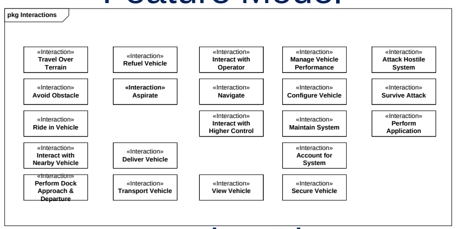
# Vehicle Pattern Example: SysML Model, Organized in Packages



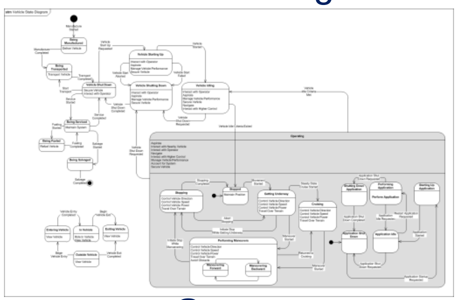
Domain Model



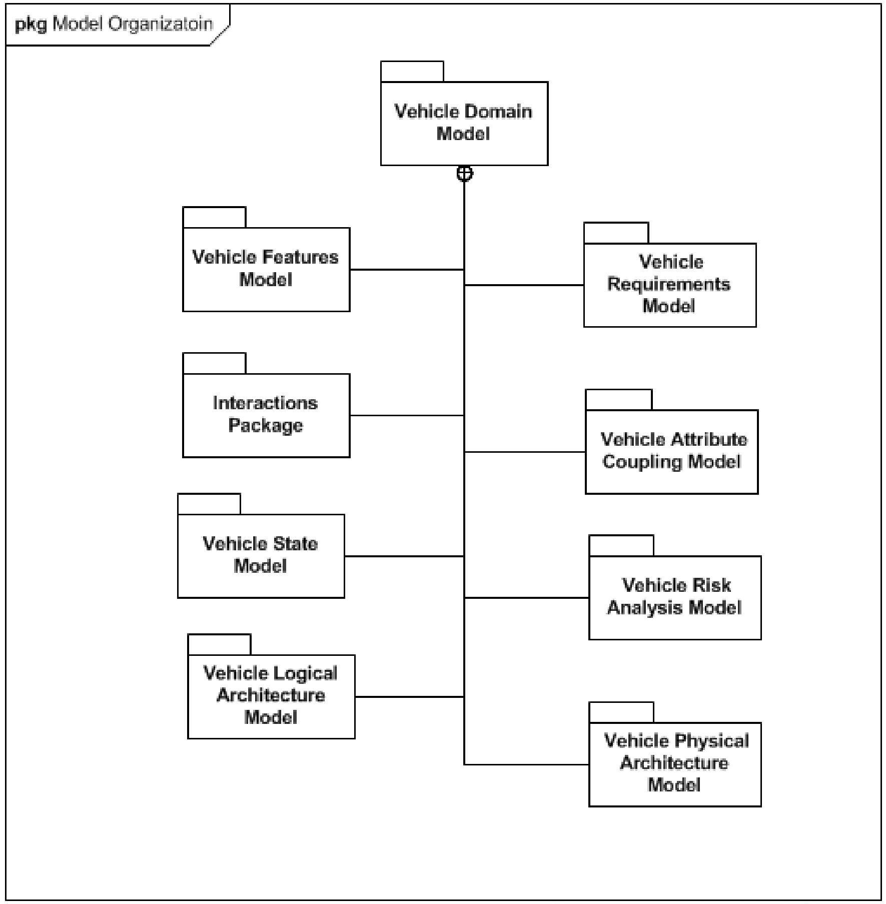
Feature Model

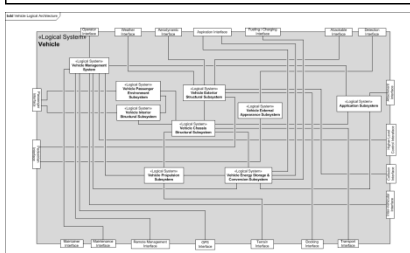



Interaction Pkgs



State

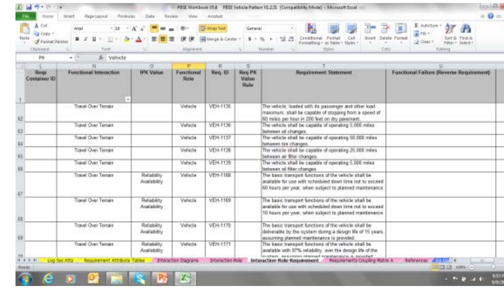


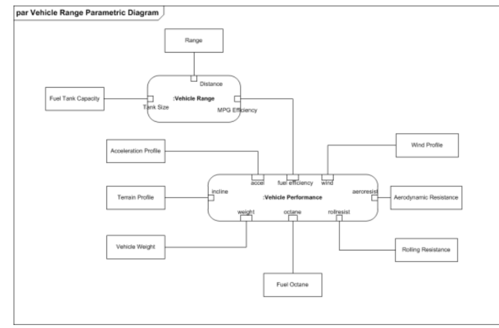
Logical Architecture

| Physical Entity | Failure Mode |
|---|---|
| Vehicle ECM | Dead ECM |
| Vehicle ECM | Network Connector Open |
| Vehicle ECM | Network Connector Short |
| Vehicle ECM | Erratic ECM |
| Battery | Discharged Battery |
| Battery | Battery Cell Short |
| Battery | Battery Cell Open |
| Battery | Battery Leak |
| Panel Display | Fractured Display |
| Panel Display | Illuminator Fail |
| Bluetooth Module | Module Hard Fail |
| Bluetooth Module | Transmitter Fail |
| Bluetooth Module | Receiver Fail |

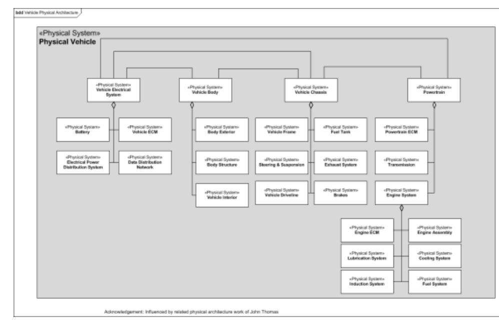
FMEA



Integrated Tables/Models




Parametrics



Physical Architecture

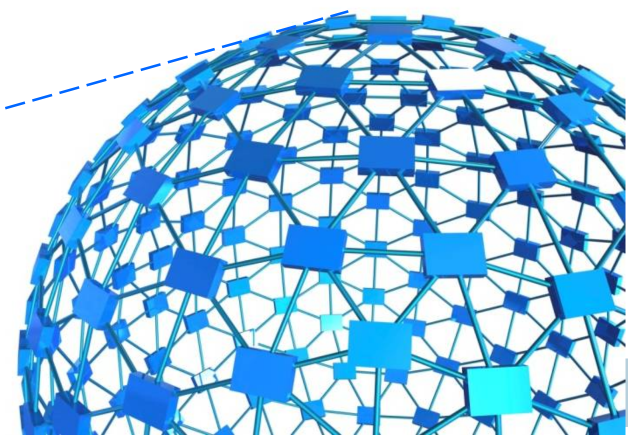
# Status of PBSE

– The major aspects of PBSE have been defined and practiced for years across a number of enterprises and domains, but with limited INCOSE community awareness:

| | | | |
|---|---|---|---|
| Medical Device Patterns | Construction Equipment Patterns | Commercial Vehicle Patterns | Space Tourism Pattern |
| Manufacturing Process Patterns | Vision System Patterns | Packaging System Patterns | Lawnmower Pattern |
| Embedded Intelligence Patterns | Systems of Innovation (SOI) Pattern | Baby Product Pattern | Orbital Satellite Pattern |
| Development Process Patterns | Production Material Handling Patterns | Engine Controls Patterns | Military Radio Systems Pattern |

– This talk is more about INCOSE community awareness and capability than about technically establishing a new method—although it will look new to INCOSE practitioners.

– We recognize that the human change aspect can be the most challenging – but are not suggesting that we also have to create new technical methods. We are introducing PBSE to a larger community.



# INCOSE Patterns Challenge Team

- Wave 1 Projects (completed in 2014):
  - Using Patterns in Automated Verification of Safety-Critical Systems
  - Patterns for Reducing Error Escapes in Development
  - Life Cycle Patterns Across the Enterprise
  - Automated Vehicle Pattern
  - The Case for a Stronger Foundation Metamodel for MBSE: Parts 1, 2
- Wave 2 Projects (2014-2015): (in progress)
  - Agile Systems Engineering Life Cycle Pattern (Joint with Agile WG)
  - SE Community Social Network Pattern
  - INCOSE Summary of PBSE Methodology
  - The Case for a Stronger Foundation Metamodel for MBSE: Part 3
- Future Projects: (2015-____)
  - Your interests? Suggest! Question! Join us!
- Co-chairs: Bill Schindel schindel@ictt.com
  Troy Peterson peterson_troy@bah.com
- Team web page (in INCOSE/OMG MBSE wiki):
  http://www.omgwiki.org/MBSE/doku.php?id=mbse:patterns:patterns

# Agile Systems Engineering Life Cycle Model (ASELCM) Project

- INCOSE sponsored project, announced at IW2015—supporting Agile and SE objectives.
- Project discovery workshops will occur at participating host company/institution sites across U.S. and Europe during 2015-2016.
- Learnings will be organized using the ASELCM Pattern—part of the project report.
- You and your organization are invited to participate.
- See: http://www.parshift.com/ASELCM/Home.html

# Challenges and Opportunities: Human hurdles

- Engineers and other designers enjoy creating things—sometimes even if the thing has been created before:
  - This may lead to re-traveling paths, sometimes re-discovering things the hard way
  - In any case, it can expend time and effort in re-generating, re-validating, and re-verifying what others had already done.

- In other cases, human subject matter experts provide great expertise:
  - but it is accessible only in the form of the presence of the SME, and after accumulating years of experience.
  - Seemingly more a craft of journeymen experts than a discipline based upon teachable scientific principles.

- All these challenges can be viewed as resistance to expressing and applying explicit patterns.

# Conclusions

1. Patterns abound in the world of systems engineering.
2. These patterns extensively impact our projects, whether we take advantage of them as Explicit Patterns, or we are negatively impacted by Dark Patterns.
3. Pattern-Based Systems Engineering (PBSE) offers specific ways to extend MBSE to exploit Patterns.
4. MBSE comes first—Patterns without Models is like orbital mechanics before Newton.
5. PBSE provides a number of identified benefits.
6. We've had good success applying pattern-based methods in mil/aerospace, automotive, medical/health care, advanced manufacturing, and consumer product domains.
7. INCOSE provides good PBSE opportunities to "learn by doing" – join us!

# References


***Representing Systems:***

1.W. Schindel, "Requirement statements are transfer functions: An insight from model-based systems engineering", *Proceedings of INCOSE 2005 International Symposium*, (2005).

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

3. B. Van Fraassen, *Scientific Representation: Paradoxes of Perspective*, Oxford U Press, 2008.

***Patterns; Pattern-Based Systems Engineering:***

4.INCOSE MBSE Methodology Summary: Pattern-Based Systems Engineering (PBSE), Based On S*MBSE Models, V1.3.2, INCOSE Patterns Challenge Team, January, 2015, at http://www.omgwiki.org/MBSE/doku.php?id=mbse:patterns:patterns_challenge_team_mtg_01.26-27.15

5.INCOSE Patterns Challenge Team of MBSE Initiative web site, with link to downloadable refs: http://www.omgwiki.org/MBSE/doku.php?id=mbse:patterns:patterns

6.W. Schindel, T. Peterson, "Introduction to Pattern-Based Systems Engineering: Leveraging MBSE", INCOSE IS2013 Tutorial, Philadelphia, 2013.

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

8.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).

9.W. Schindel, "The Impact of 'Dark Patterns' On Uncertainty: Enhancing Adaptability In The Systems World", INCOSE Great Lakes 2011 Conference, Dearborn, MI, 2011.

10.Eric Berg, "Affordable Systems Engineering: An Application of Model-Based System Patterns to Consumer Packaged Goods Products, Manufacturing, and Distribution", INCOSE IW2014 MBSE Workshop, Los Angeles, CA, January 2014.

11.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.

12.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.

***PBSE, continued:***

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

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

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

16.D. Williams, "How Concepts of Self-Regulation Explain Human Knowledge", *The Bent of Tau Beta Pi*, Winter (2011).

17.Kahneman, D., *Thinking, Fast and Slow*, Farrar, Straus and Giroux, Publishers, 2011, ISBN-10: 0374275637.

18.Lewis, Michael, *Moneyball: The Art of Winning an Unfair Game*, Norton, New York, 2004.

***Agile Systems Engineering*** **:**

19.Rick Dove, Ralph LaBarge, "Fundamentals of Agile Systems Engineering—Part 1" and "Part 2", Proceedings of INCOSE IS2014, July, 2014.

20.W. Schindel, "Introduction to the Agile Systems Pattern: An MBSE-Based System Pattern, with Implications for Agile Modeling", INCOSE IW2015 MBSE Workshop, Jan, 2015, at http://www.omgwiki.org/MBSE/doku.php?id=mbse:incose_mbse_iw_2015:breakout_out_session_agile_modeling

21.INCOSE ASELCM Project Web Site: http://www.parshift.com/ASELCM/Home.html

***Other Systems Engineering References*****:**

22.ISO/IEC 15288: Systems Engineering—System Life Cycle Processes. International Standards Organization (2008).

*23.INCOSE Systems Engineering Handbook: A Guide for System Life Cycle Processes and Activities*, Version 3.2, International Council on Systems Engineering (2010).

*24.NASA Systems Engineering Handbook,* NASA/SP-2007-6105, Rev 1, U.S. National Aeronautics and Space Administration (2007).

25.W. Schindel, "Failure Analysis: Insights from Model-Based Systems Engineering", *Proceedings of INCOSE 2010 Symposium*, July 2010.

# About the presenters

Troy Peterson is a Chief Engineer and Fellow at Booz Allen Hamilton and his expertise is in strategy, systems engineering and management. He has led several distributed teams in delivery of large-scale complex systems and has instituted numerous organizational processes to improve efficiency and effectiveness. His consulting experience spans academic, commercial and government sectors as well as all lifecycle phases of program and product development. Troy completed advanced graduate studies at Massachusetts Institute of Technology in System Design and Management, obtained a MS in Business and Technology Management from Renesslaer Polytechnic Institute and BS in Mechanical Engineering from Michigan State University, and.  Troy is also the Past President of the INCOSE Michigan Chapter and an INCOSE CSEP, PMI PMP, and ASQ CSSBB.

William D. (Bill) Schindel is co-chair of the System Patterns Challenge Team (a part of the INCOSE/OMG MBSE Initiative), and co-lead of the INCOSE Agile Systems Engineering Life Cycle Model Project, announced at IW2015.  His forty-year engineering career has included aerospace engineering with IBM Federal Systems, teaching engineering and mathematics at Rose-Hulman Institute of Technology, founding and leading a supplier of telecom carrier network control systems for the public network, and leading ICTT System Sciences (www.ictt.com), a systems engineering enterprise that has pioneered Pattern-Based Systems Engineering methods for transforming the productivity of the innovation process in medicine and health care, advanced manufacturing, aerospace, automotive, and consumer products.  Bill is an INCOSE CSEP and president of the Crossroads of America INCOSE chapter.

# Abstract

Abstract: This tutorial is a brief overview of Pattern-Based Systems Engineering (PBSE), including some specific system domain illustrations.

INCOSE 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 INCOSE currently reaches. The PBSE Challenge Team of the INCOSE/OMG MBSE Initiative aims to enable INCOSE membership, and the larger systems community beyond INCOSE, to achieve such order-of-magnitude improvements.

PBSE leverages the power of Model-Based Systems Engineering (MBSE) to rapidly deliver benefits to a larger community. Projects using PBSE get a "learning curve jumpstart" from an existing Pattern, gaining the advantages of its content, and improve that pattern with what they learn, for future users. The major aspects of PBSE have been defined and practiced some years across a number of enterprises and domains, but with only limited INCOSE community awareness.