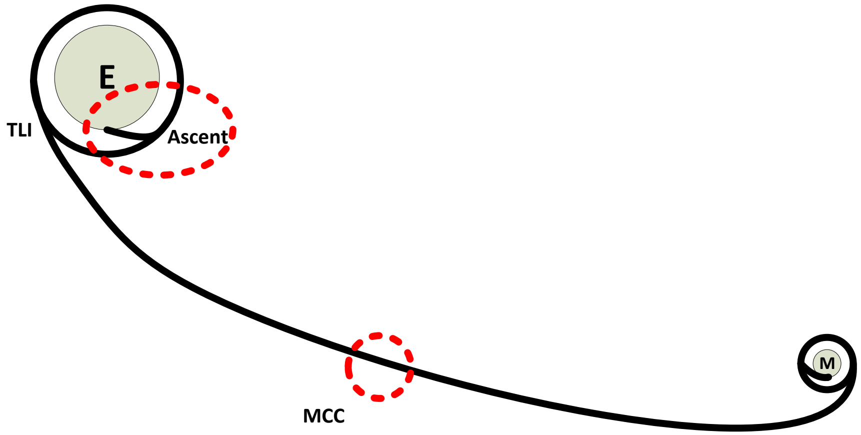
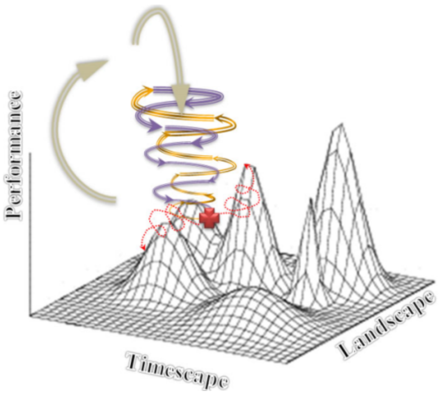
25th anniversary
annual INCOSE
international symposium
Seattle, WA
July 13 - 16, 2015



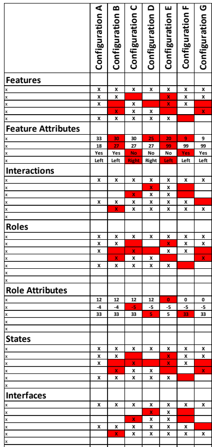
System Life Cycle Trajectories

Tracking Innovation Paths Using System DNA

Bill Schindel, ICTT System Sciences
schindel@ictt.com



1.6.1



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

Abstract

In-service systems change configuration, across their life cycles. Systems in development change in-progress developmental configurations. Evolving product lines and competing product models change configuration, across the introduction of new entrants.

Understanding trajectories (paths of changing configurations) of systems is important to understanding installation history, developmental progress, or competitive evolution. This is true whether we are interested in an in-service system, a system still in development, or an ensemble of product systems evolving over time as they compete. The traditional discipline of Configuration Management (CM) and the emerging automation aids of Product Lifecycle Management (PLM) provide important help in addressing this challenge. However, as system complexity and rate of change continue to grow, it is not clear that these tools in their current forms are sufficient to support fully understanding the trajectories of systems. This is particularly the case when a premium is placed on accelerating the rate of system evolution and decision-making in competitive markets. There is no lower non-zero bound to the shrinking response times demanded by such competition, as illustrated by agile, composable systems and security challenges.

The development of microbiology offers some insights. Whereas the physical form (phenotype) of living systems was the historical focus in understanding their configuration and its evolution, today the study of evolution of genetic information (genotype) is a vital part of that understanding. Growth of understanding of the genetic basis of life has driven this shift.

This presentation reports on a model-based representation of trajectories of “system DNA”, using MBSE and PBSE. The ideas spring from work reported by a System Sciences Working Group project during IW2014, and address issues of current interest to the Model Management Working Group. Those responsible for configuration of systems or families will learn about effective ways to represent trajectories.

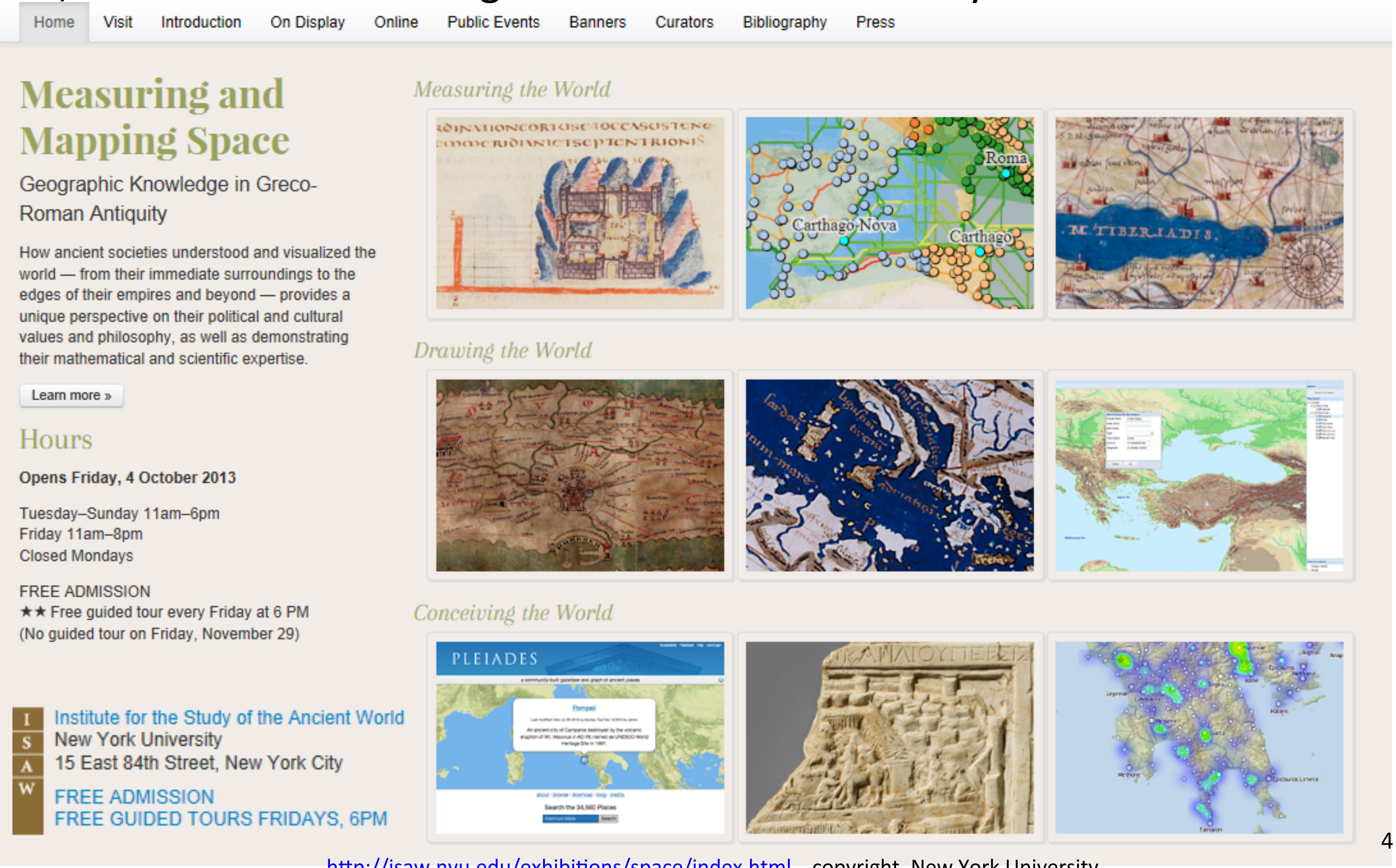
Contents

- Maps and itineraries of the ancient navigator
- Maps and itineraries of the systems engineer
- Stronger semantic models of system configuration space
- Trajectories in system configuration space
- Why trajectories are important: Agility in innovation
- Persistent memory: Accumulation of experience is essential for agility
- Implications for Systems Engineering

- References

Maps and Itineraries of Ancient Navigators

- Scholars[4,5] suggest that ancient (Greco-Roman) navigators did not possess the "ancient maps" attributed to them—they were produced in later ages!
- So, what did ancient navigators use to find their way?



Measuring and Mapping Space

Geographic Knowledge in Greco-Roman Antiquity

How ancient societies understood and visualized the world — from their immediate surroundings to the edges of their empires and beyond — provides a unique perspective on their political and cultural values and philosophy, as well as demonstrating their mathematical and scientific expertise.

Hours

Opens Friday, 4 October 2013

Tuesday–Sunday 11am–6pm
Friday 11am–8pm
Closed Mondays

FREE ADMISSION
★★ Free guided tour every Friday at 6 PM
(No guided tour on Friday, November 29)

Measuring the World

Drawing the World

Conceiving the World

ISAW

Institute for the Study of the Ancient World
New York University
15 East 84th Street, New York City

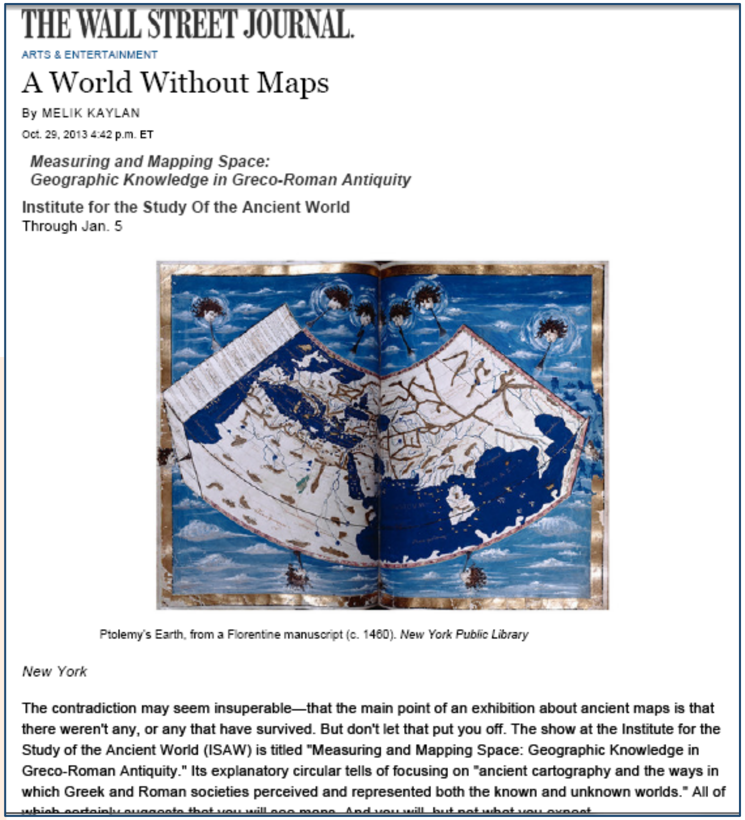
FREE ADMISSION
FREE GUIDED TOURS FRIDAYS, 6PM

http://isaw.nyu.edu/exhibitions/space/index.html

From: “A World Without Maps”

- *Wall Street Journal,* 10.30.2013 [4]
- The exhibition at the Institute for the Study of the Ancient World [5]
- It demonstrates that what we think of as “ancient maps” were created long after the period in which we assume they were used, so the reviewer asks . . .

“Why do we have virtually no ancient maps of the ancient world? After all, sailors, traders and soldiers had to find their way around. The show's curator, Roberta Casagrande-Kim, distinguishes between a map and an itinerary. The latter ‘must have existed aplenty, but being strictly functional probably deteriorated through overuse,’ she says. ‘A map, however small its focus, suggests a kind of implicit overview, and that is the show's subject.’” *(emphases added)*



THE WALL STREET JOURNAL.

ARTS & ENTERTAINMENT

A World Without Maps

By MELIK KAYLAN

Oct. 29, 2013 4:42 p.m. ET

Measuring and Mapping Space: Geographic Knowledge in Greco-Roman Antiquity

Institute for the Study Of the Ancient World
Through Jan. 5

Ptolemy's Earth, from a Florentine manuscript (c. 1460). *New York Public Library*

New York

The contradiction may seem insuperable—that the main point of an exhibition about ancient maps is that there weren't any, or any that have survived. But don't let that put you off. The show at the Institute for the Study of the Ancient World (ISAW) is titled "Measuring and Mapping Space: Geographic Knowledge in Greco-Roman Antiquity." Its explanatory circular tells of focusing on "ancient cartography and the ways in which Greek and Roman societies perceived and represented both the known and unknown worlds." All of which certainly suggests that you will see maps. And you will, but not what you expect.

Itineraries are not Maps

- “Greeks and Romans usually employed what are known as *periploi* (‘coastal navigations’), which *listed* ports and landmarks to facilitate commercial and military sailing, and *itineraria* (“journeys”), *lists* of locations and distances based on land routes.”
    - from: “Measuring and Mapping Space: Geographic Knowledge in Greco-Roman Antiquity” (NYU ISAW)

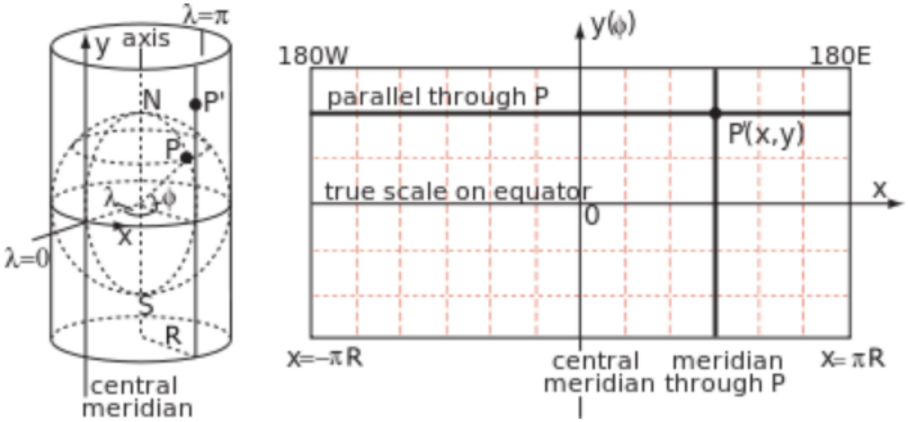
Itinerary ≠ Map!

(What am I doing?) *(Where am I?)*

When they eventually did emerge, maps represented a newer idea of the nature of “where”.

Patterns of Thought:
Maps are More than Artifacts

- A key point of these scholars is that the ancient navigators lacked more than the physical map artifacts:
  - They had not yet developed the *<u>mental paradigms</u>* associated with later emergence of geographic maps. [5,6]



- A better known example: The later Mercator projection of sphere onto cylinder.

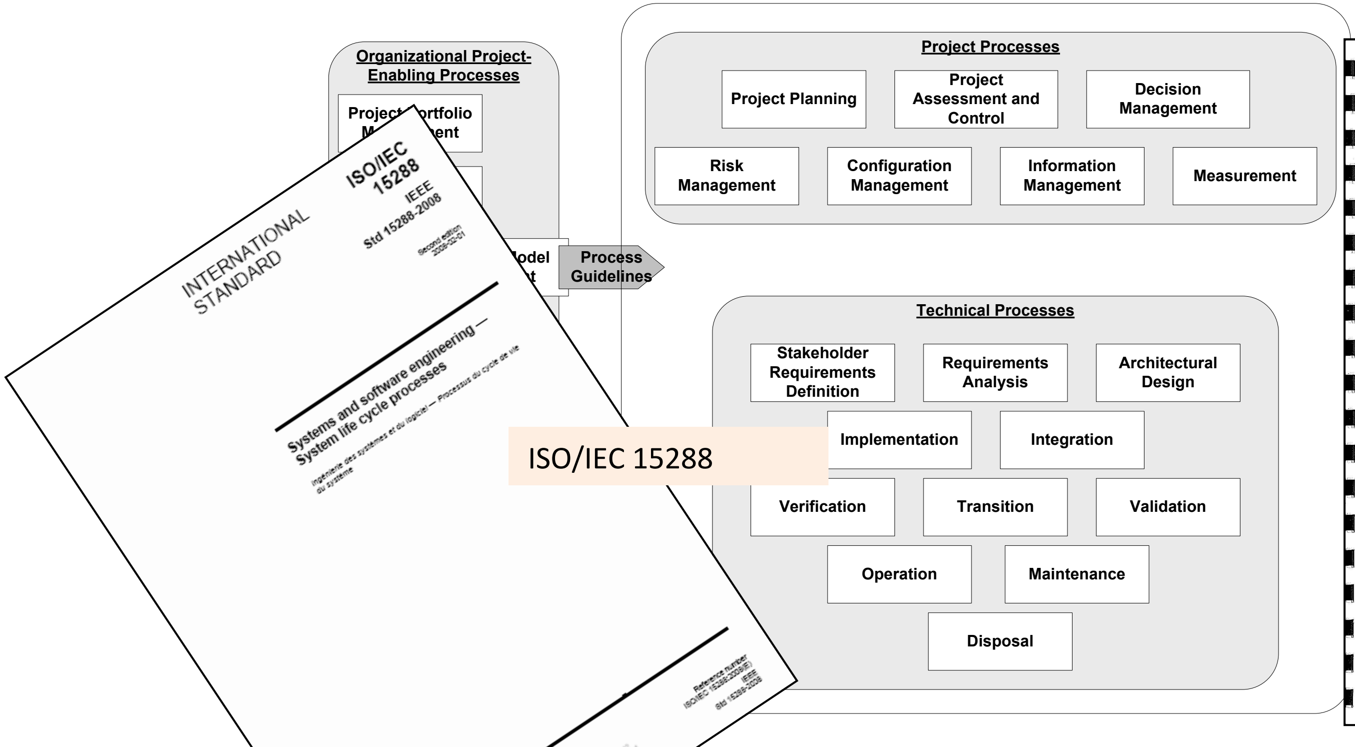
Maps and Itineraries of the Systems Engineer

- Systems Engineers must "navigate" a different type of "journey"—a *project*:
  - More complex and abstract than physical travel
  - But, it still has a starting point and a destination
  - With opportunities to become lost or disoriented
  - With risks of not reaching the desired destination
- Is this more than just a metaphorical comparison?
  - Yes: We will argue that it can be much more!

Maps and Itineraries of the Systems Engineer

- Systems Engineers have plenty of “itineraries” to guide their work, in the form of processes and procedures:
  - International Standards
  - Professional Society and Trade Group Publications
  - Enterprise-specific processes and procedures

Corporate Processes, Procedures



Organizational Project-Enabling Processes

Project Processes

Project Planning

Project Assessment and Control

Decision Management

Risk Management

Configuration Management

Information Management

Measurement

Process Guidelines

Technical Processes

Stakeholder Requirements Definition

Requirements Analysis

Architectural Design

Implementation

Integration

Verification

Transition

Validation

Operation

Maintenance

Disposal

INTERNATIONAL STANDARD

ISO/IEC 15288

IEEE Std 15288-2008

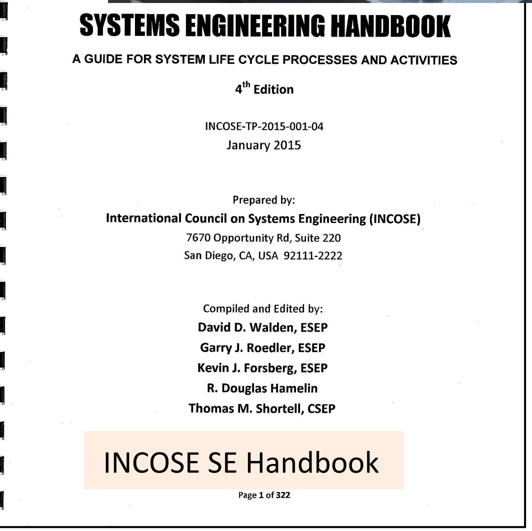
Second edition 2008-02-01

Systems and software engineering — System life cycle processes

Ingénierie des systèmes et du logiciel — Processus du cycle de vie du système

Reference number ISO/IEC 15288:2008(E) IEEE Std 15288-2008

ISO/IEC 15288



SYSTEMS ENGINEERING HANDBOOK

A GUIDE FOR SYSTEM LIFE CYCLE PROCESSES AND ACTIVITIES

4th Edition

INCOSE-TP-2015-001-04
January 2015

Prepared by:
International Council on Systems Engineering (INCOSE)
7670 Opportunity Rd, Suite 220
San Diego, CA, USA 92111-2222

Compiled and Edited by:
David D. Walden, ESEP
Garry J. Roedler, ESEP
Kevin J. Forsberg, ESEP
R. Douglas Hamelin
Thomas M. Shortell, CSEP

Page 1 of 322

INCOSE SE Handbook

Maps and Itineraries of the Systems Engineer

- Have you ever witnessed this problem?
  - The junior engineer says he has done all the steps.
  - All the checklist boxes are checked.
  - But the result is not acceptable.

Maps and Itineraries of the Systems Engineer

- It is clear what an SE Itinerary is, but what is an SE map?
  - SE Map: is not a list of SE tasks. Descriptions of systems work (Vee diagrams, ISO/IEC 15288, INCOSE SE Handbook, enterprise business procedures, etc.) are closer to itineraries than to maps.
  - SE Map: is not a model of the process--ancient mariners where not traveling through "step space", but through *geographic space*.
  - A geographic map describes *where* we really want to end up, along with key relationships around it, in 1, 2, or 3 dimensions (degrees of freedom in geographic space), and *where* we are along the way.
  - Knowing steps we have performed does not guarantee "location" (dead reckoning).
  - So, what is an "SE Map"?

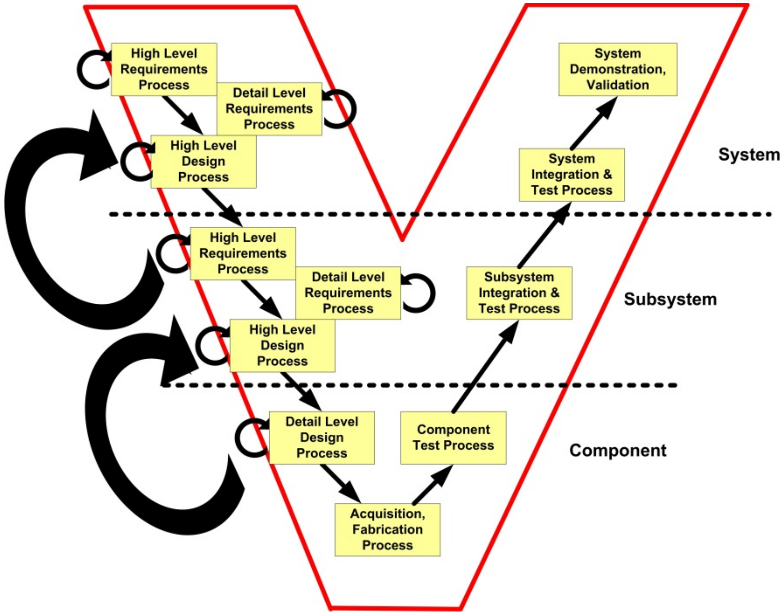
Itinerary ≠ Map!

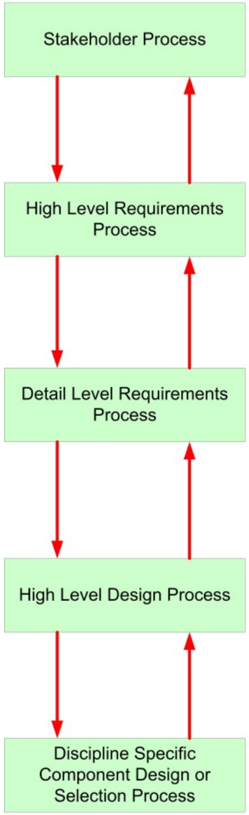
(What should I do?)

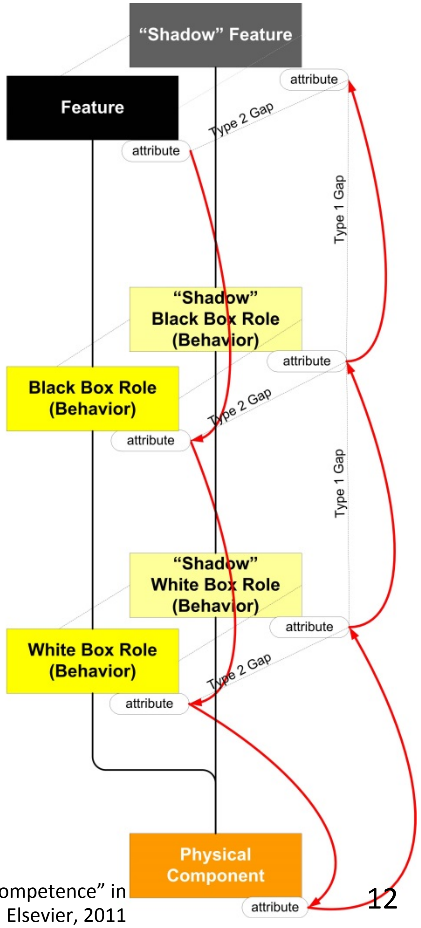
(Where am I?)

(Where am I going?)

- A practical connection is this --
  - Since the innovation cycle is inherently iterative, ...
  - How do we know when we are "done"?
  - It is not by knowing what steps we have completed, ...
  - It is by knowing how "close" our current configuration is to the "destination" we are seeking.
  - The distance metrics are in *configuration space*.


High Level Requirements Process
Detail Level Requirements Process
High Level Design Process
System Demonstration, Validation
System Integration & Test Process
System
High Level Requirements Process
Detail Level Requirements Process
High Level Design Process
Subsystem Integration & Test Process
Subsystem
Detail Level Design Process
Component Test Process
Component
Acquisition, Fabrication Process


Stakeholder Process
High Level Requirements Process
Detail Level Requirements Process
High Level Design Process
Discipline Specific Component Design or Selection Process

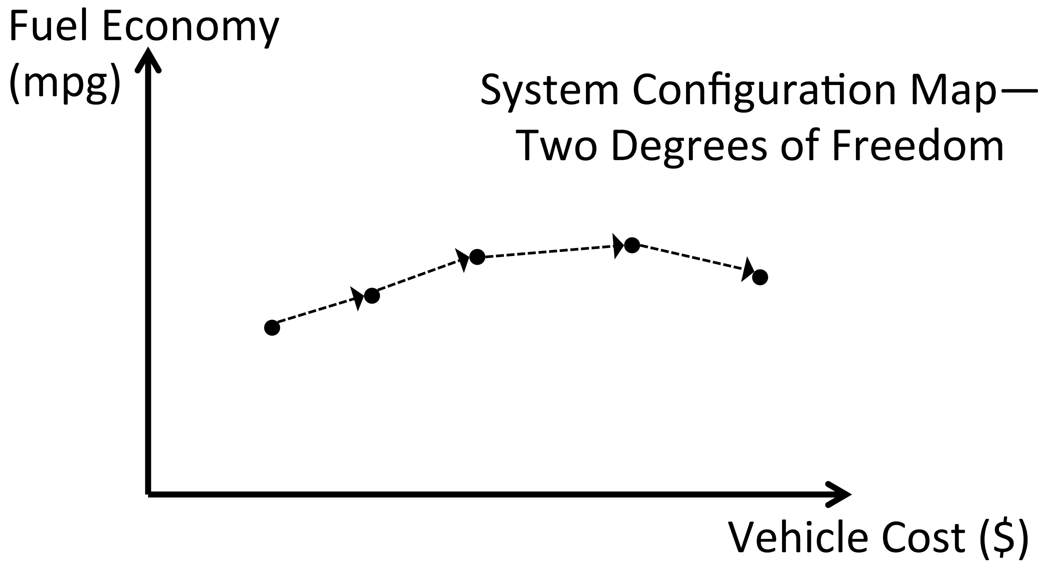


From: W. Schindel, "Innovation as Emergence: Hybrid Agent Enablers for Evolutionary Competence" in *Complex Adaptive Systems, Volume 1*, Cihan H. Dagli, Editor in Chief, Elsevier, 2011

Maps and Itineraries of the Systems Engineer

- The work of engineering is performed on, and produces, information.
- A map appropriate to this territory would be a map about that information—not the steps of a procedure (the itinerary) processing it.
- We know one kind of "map about information": an information model (i.e., E-R model)
- The hard sciences (laws) provide the underlying relationship map for physics, chemistry, etc. This is why their related engineering practices (mechanical, electrical, chemical engineering) are able to navigate more generally.
- Imagine trying to learn chemistry by studying the process of cooking instead of studying the materials in process!
- The SE Map describes the system configuration space of possible places to be, good and not good, and how they are related to each other.
- Early "systems engineering" itineraries (still dominant!) are not maps through the information navigated by those procedures.
- As we begin making real information model maps of the information, there are many startling and valuable discoveries.

Simple Example of a Trajectory on a System Map: Two Degrees of Freedom



- Of course, we'd likely add many more degrees of freedom (weight, range, etc.)—so system maps will tend to be high dimension, and subject to "slicing" into multiple views.
- During innovation / development cycles, and some life cycles, the "current configuration" may involve sets of ranges or lists, instead of individual points, so the trajectory becomes an ordered series of envelopes.

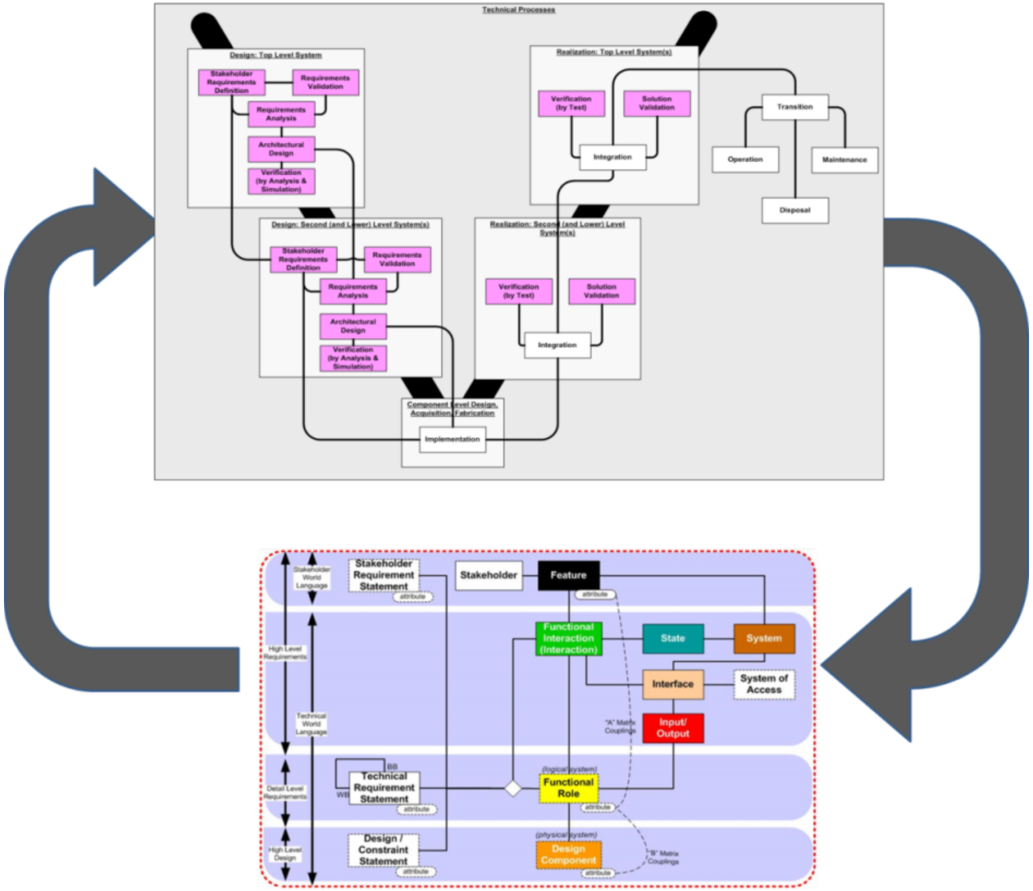
Moving to Stronger Semantic Models of Systems

- "System Configuration Space" is the multi-dimensional space, in which each "point" represents one possible configuration of a system of interest.
- A "trajectory" through this space is a set of system configurations, "visited" as the configuration of the (modeled) system is changed.
- The different degrees of freedom of this space are related to each other, by a system model.
- Such a system model may be expressed using a system modeling language, such as SysML, covering enough variables and relationships to describe the system for SE purposes.
- Model-Based Systems Engineering (MBSE) is growing in popularity, but procedure is still the dominant way people think about systems engineering, even with model-based artifacts, but this is shifting . . .

Moving to Stronger Semantic Models of Systems

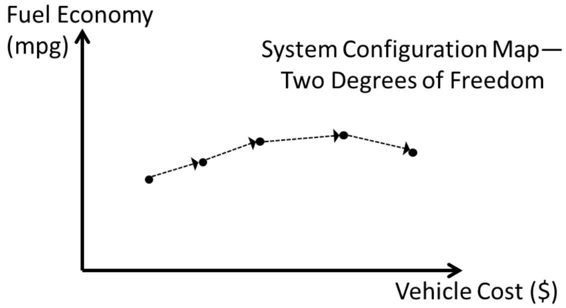
A start is to view the engineering model as what passes through the engineering process, in a series of transformations:

Innovation Process



Information Passing Through the Innovation Process

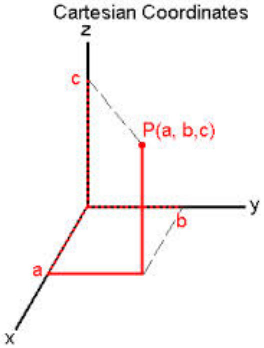
<u>SE Process:</u> For example, modeled as ISO15288 process areas.



*<u>S*Trajectory</u> as a series of system configurations in S*Configuration Space, through iterations of the SE process:*

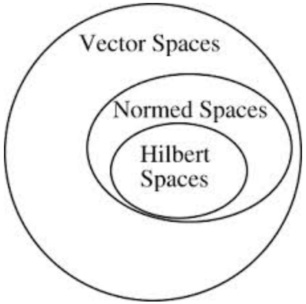
System Space: The Geometrization of System Models

- Such a geometric shift in thinking (about spaces of systems) is reminiscent of earlier geometric shifts in human thinking:
  - Geometrization of algebra, by Rene Descartes ("Cartesian" coordinates):



Rene Descartes
1596 - 1650

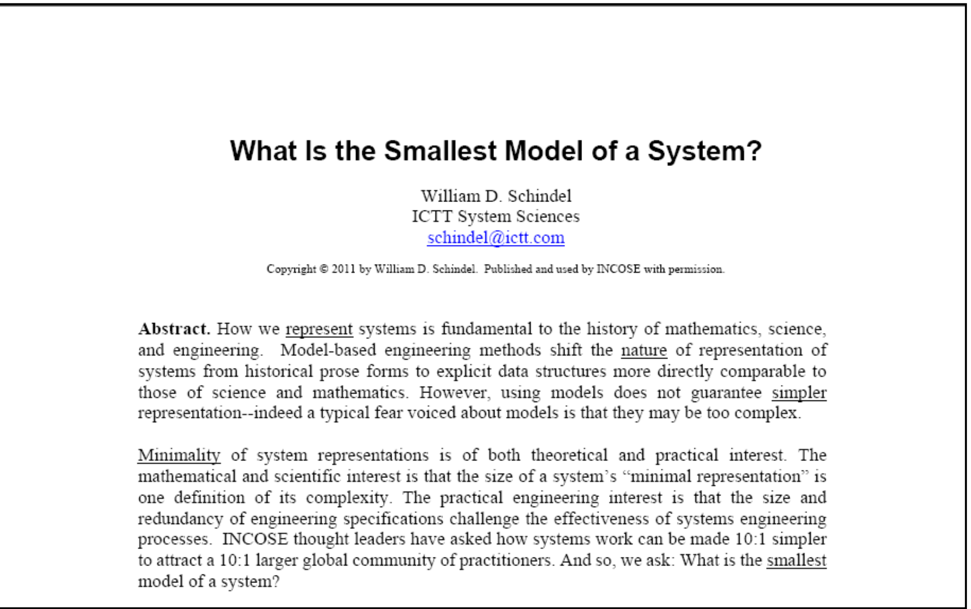
- Just as system models also add modeling of (infinite dimensional) behavior, Hilbert Space (David Hilbert) provided the next required generalization, supporting a geometrical view of mathematical function:



David Hilbert
1862 - 1943

- Geometrization of mathematical models does not ultimately mean drawing geometric diagrams (as in 2D & 3D geographic maps), but instead provides geometry-based intuitive basis for more abstract mathematical concepts: distance (metric spaces), projections, inner products, paths in configuration space.

- What are the degrees of freedom (variables) needed by System Models?
  - System modeling languages (SysML, OPM, IDEF, etc.) have progressed; however . . .
  - At least some thought leaders agree that these models are more syntactical than semantic, with none of them currently a complete semantic model of the subject systems.[3]
  - Too big and too small at the same time.
  - What is the Smallest Model of a System? [1,2]

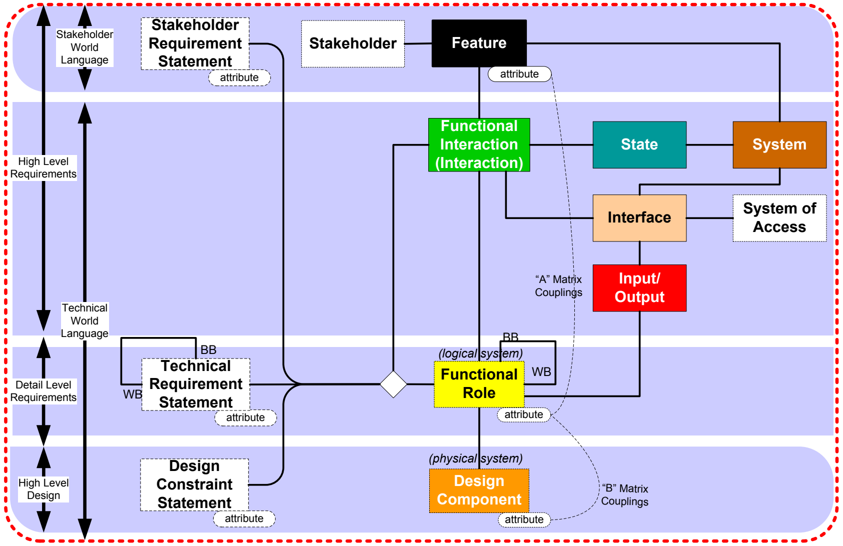


What Is the Smallest Model of a System?

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?

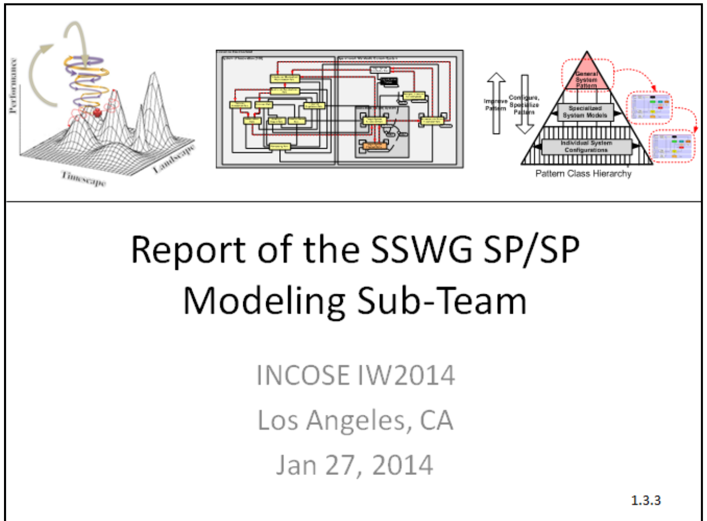


Moving to Stronger Semantic Models of Systems

- Why is such a transition in thought important?
  - Because of what happened in science, engineering & mathematics after the relationships were discovered and became explicit.
  - Relational clues from the history of physical sciences.
  - Prime example: the central role of physical interactions as the basis of all scientific law.
- INCOSE Vision 2025 envisions this kind of progress
- But first, models of systems must achieve some improvement to their foundations:
  - Stronger semantic metamodel in MBSE.
  - Difference between modeling business process information about systems and views, versus modeling the systems themselves, in the tradition of science.

Sufficient representations of system configuration and trajectories

- What are those N degrees of freedom? What are the variables? How shall we view them?
  - S*Models: The smallest model of a system, for purposes of engineering or science.
  - Illustrated / reported on by the INCOSE System Science Working Group Modeling Sub-team, at IW2014: [38]



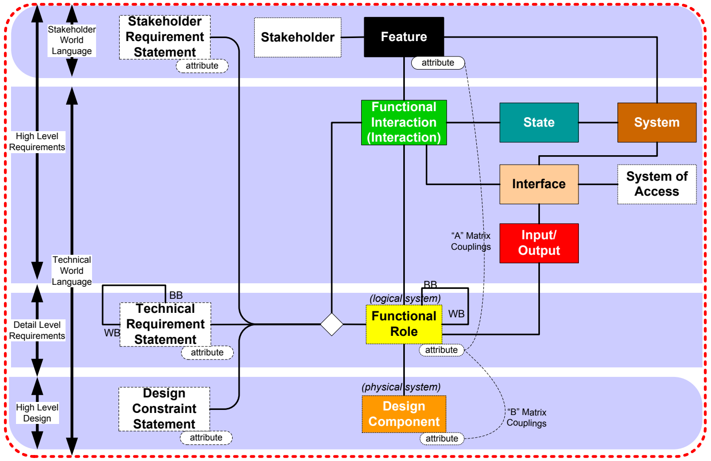
Report of the SSWG SP/SP Modeling Sub-Team

INCOSE IW2014
Los Angeles, CA
Jan 27, 2014

1.3.3

Sufficient representations of system configuration and trajectories

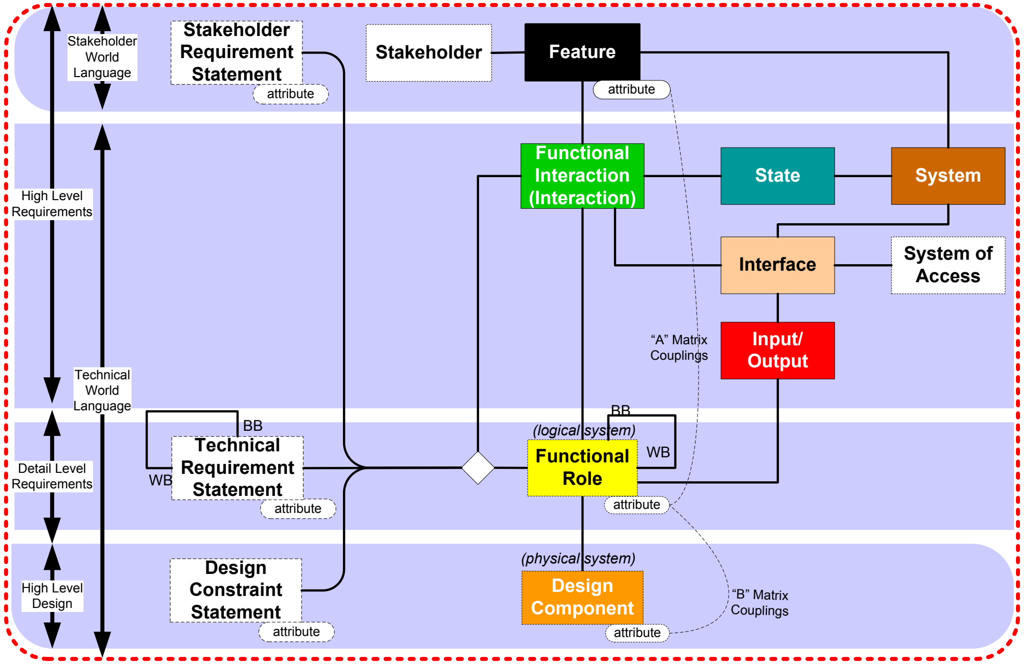
- The S*Metamodel describes system configuration space as including a number of different dimensional subspaces:
  - Stakeholder Features and their Attributes
  - External Domain Interactions, Actors, Input-Outputs, and Interfaces
  - Functional Roles and their Attributes
  - States (Modes)
  - Requirements
  - Physical Components and their Attributes
  - Failure Modes and Impacts
  - Attributes and their Value Couplings
  - Relationships between all of these
  - Others



- Instances of combined configurations of these are points in S*Configuration Space—the space of total configurations of systems.
- This leads to views of this "System DNA", and expressions of trajectories across it, in compressed form . . . .

The S*Metamodel describes system configuration space as including a number of different dimensional subspaces:

- Stakeholder Features and their Attributes
- External Domain Interactions, Actors, Input-Outputs, and Interfaces
- Functional Roles and their Attributes
- States (Modes)
- Requirements
- Physical Components and their Attributes
- Failure Modes and Impacts
- Attributes and their Value Couplings
- Relationships between all of these
- Others

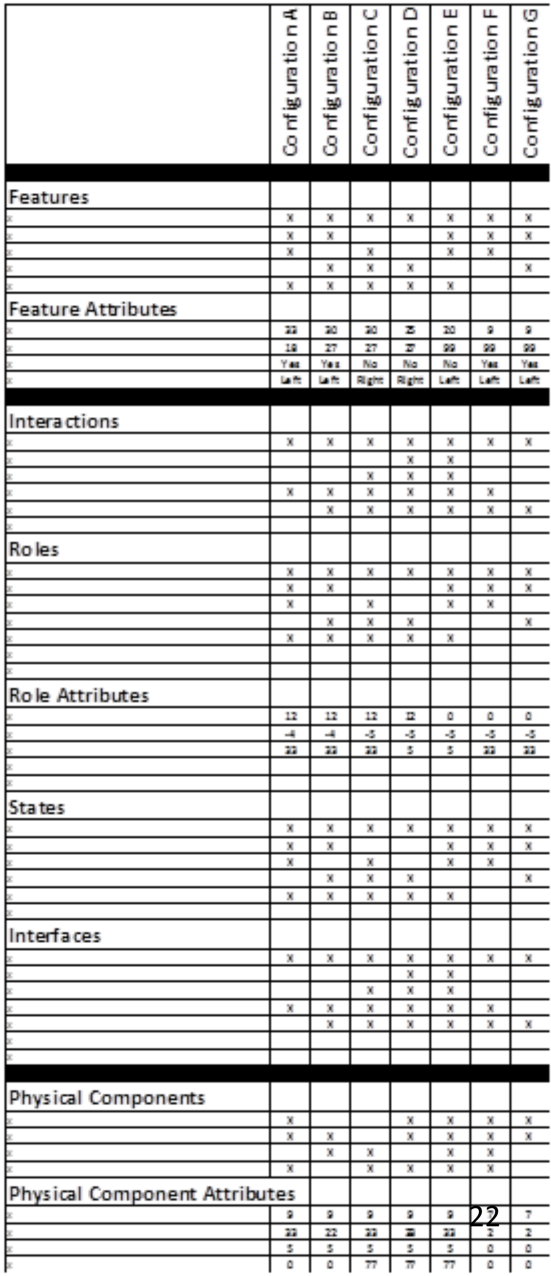


System Configuration "Genome"

Value, Fitness

Behavior

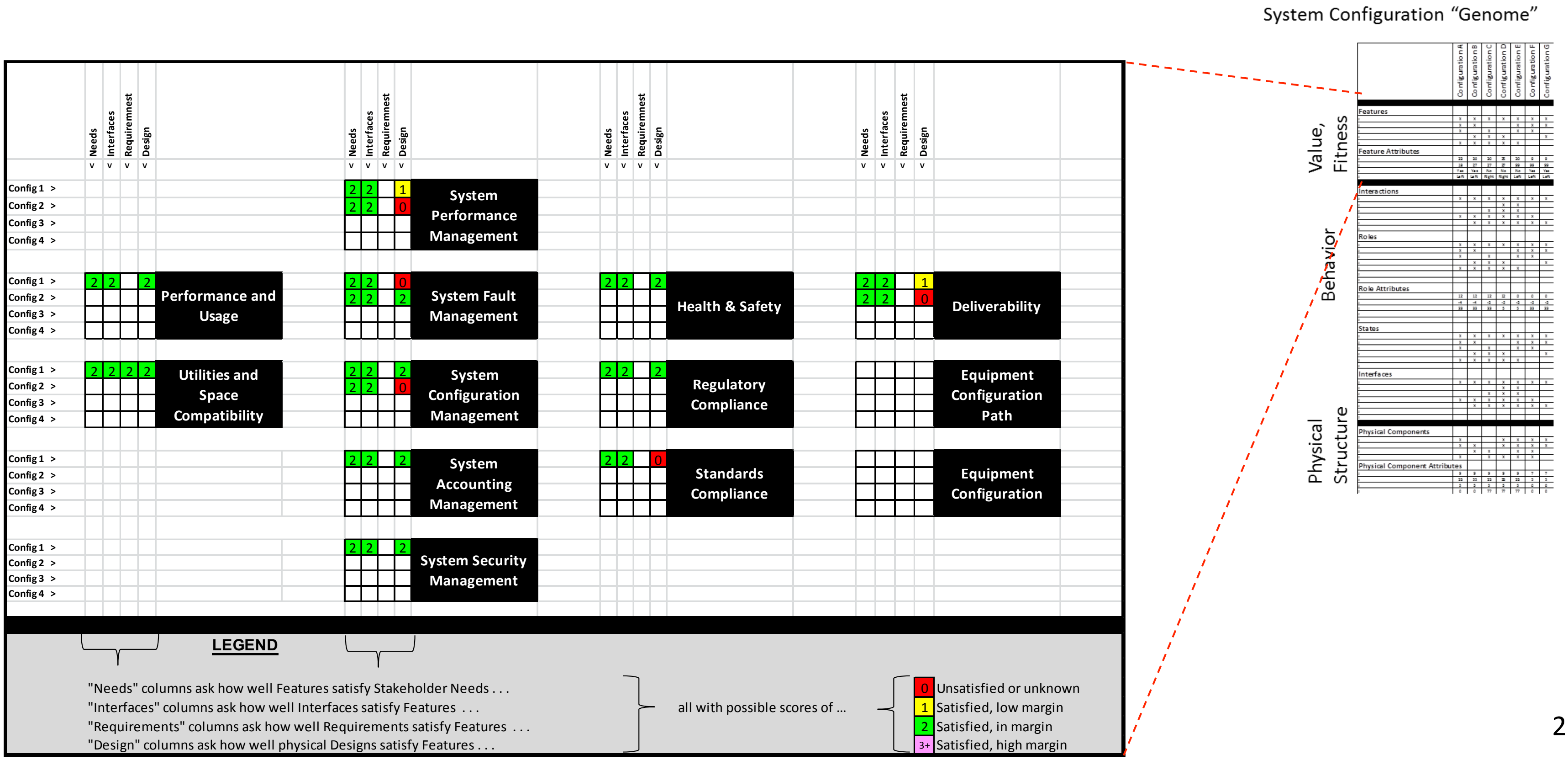
Physical Structure



	Configuration A	Configuration B	Configuration C	Configuration D	Configuration E	Configuration F	Configuration G
Features							
x	X	X	X	X	X	X	X
x	X	X			X	X	X
x	X		X		X	X	
x		X	X	X			X
x	X	X	X	X	X		
Feature Attributes							
x	22	20	20	25	20	9	9
x	18	27	27	27	99	99	99
x	Yes	Yes	No	No	No	Yes	Yes
x	Left	Left	Right	Right	Left	Left	Left
Interactions							
x	X	X	X	X	X	X	X
x				X	X		
x			X	X	X		
x	X	X	X	X	X	X	
x		X	X	X	X	X	X
x							
Roles							
x	X	X	X	X	X	X	X
x	X	X			X	X	X
x	X		X		X	X	
x		X	X	X			X
x	X	X	X	X	X		
x							
x							
Role Attributes							
x	12	12	12	12	0	0	0
x	-4	-4	-5	-5	-5	-5	-5
x	22	22	22	5	5	22	22
x							
x							
States							
x	X	X	X	X	X	X	X
x	X	X			X	X	X
x	X		X		X	X	
x		X	X	X			X
x	X	X	X	X	X		
x							
Interfaces							
x	X	X	X	X	X	X	X
x				X	X		
x			X	X	X		
x	X	X	X	X	X	X	
x		X	X	X	X	X	X
x							
x							
Physical Components							
x	X			X	X	X	X
x	X	X		X	X	X	X
x		X	X		X	X	
x	X		X	X	X	X	
Physical Component Attributes							
x	9	9	9	9	9	7	7
x	22	22	22	22	22	2	2
x	5	5	5	5	5	0	0
x	0	0	77	77	77	0	0

Stakeholder Features Subspace View Has Special Significance

- All travel through the configuration space is caused by "forces" within the feature configuration subspace:
  - Where all "whys" are represented; selection-based [31]
  - For human-engineered projects, this view is always the top level "dashboard" on progress and status
  - Highly compressible, dividing configuration vs. pattern content



LEGEND

"Needs" columns ask how well Features satisfy Stakeholder Needs . . .
"Interfaces" columns ask how well Interfaces satisfy Features . . .
"Requirements" columns ask how well Requirements satisfy Features . . .
"Design" columns ask how well physical Designs satisfy Features . . .

all with possible scores of ...

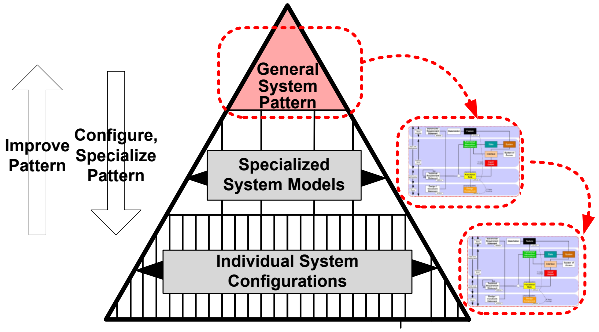
0	Unsatisfied or unknown
1	Satisfied, low margin
2	Satisfied, in margin
3+	Satisfied, high margin

“Delta” Requirements, or All Requirements?

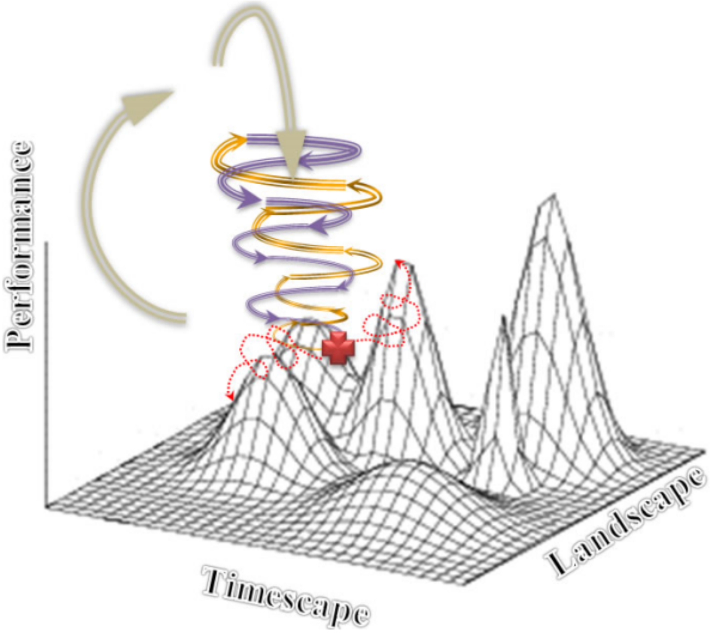
- It is very common to see specification of requirements that are “changing” in a new system version, in comparison to past history:
  - The “Delta” requirements
  - Helps call attention to what is changing and needs focal attention
- But, there are (in)famous consequences of over-emphasizing these “Delta” requirements:
  - Consequence 1: Some other aspect of the system is impacted / broken, through lack of awareness.
  - Consequence 2: Even if we don’t break anything, by going through repeated “Delta” update cycles on a series of future versions, we eventually arrive at a point where no one has a description of the complete set of requirements.
- Happily, when using a strong-enough underlying metamodel, a combined “differential + integral” form has the additional benefit of strong connection to dynamical systems of classical mechanics.

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

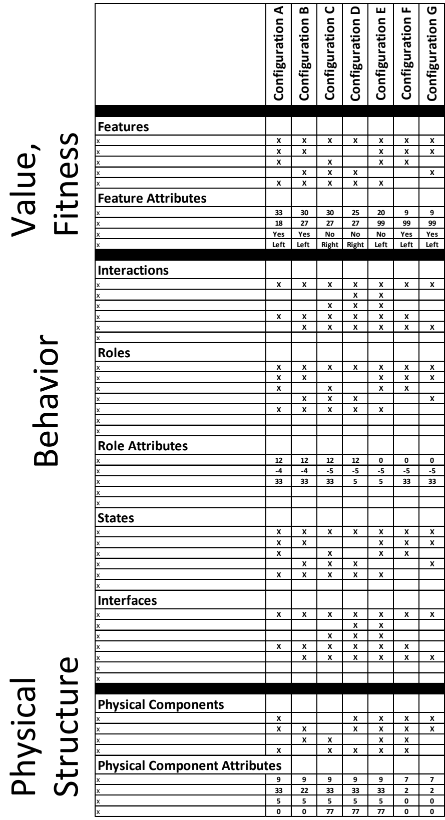
<u>Differential</u> trajectory descriptions can further compress the dimensionality of an evolutionary path.


General System Pattern
Improve Pattern
Configure, Specialize Pattern
Specialized System Models
Individual System Configurations

Pattern Class Hierarchy



System Configuration “Genome”



Value, Fitness

Behavior

Physical Structure

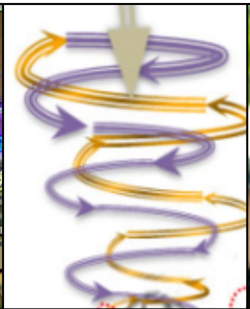
	Configuration A	Configuration B	Configuration C	Configuration D	Configuration E	Configuration F	Configuration G
Features							
x	X	X	X	X	X	X	X
x	X	X			X	X	X
x	X		X		X	X	
x		X	X	X			X
x	X	X	X	X	X		
Feature Attributes							
x	33	30	30	25	20	9	9
x	18	27	27	27	99	99	99
x	Yes	Yes	No	No	No	Yes	Yes
x	Left	Left	Right	Right	Left	Left	Left
Interactions							
x	X	X	X	X	X	X	X
x				X	X		
x			X	X	X		
x	X	X	X	X	X	X	
x		X	X	X	X	X	X
x							
Roles							
x	X	X	X	X	X	X	X
x	X	X			X	X	X
x	X		X		X	X	
x		X	X	X			X
x	X	X	X	X	X		
x							
x							
Role Attributes							
x	12	12	12	12	0	0	0
x	-4	-4	-5	-5	-5	-5	-5
x	33	33	33	5	5	33	33
x							
x							
States							
x	X	X	X	X	X	X	X
x	X	X			X	X	X
x	X		X		X	X	
x		X	X	X			X
x	X	X	X	X	X		
x							
Interfaces							
x	X	X	X	X	X	X	X
x				X	X		
x			X	X	X		
x	X	X	X	X	X	X	
x		X	X	X	X	X	X
x							
x							
Physical Components							
x	X			X	X	X	X
x	X	X		X	X	X	X
x		X	X		X	X	
x	X		X	X	X	X	
Physical Component Attributes							
x	9	9	9	9	9	7	7
x	33	22	33	33	33	2	2
x	5	5	5	5	5	0	0
x	0	0	77	77	77	0	0

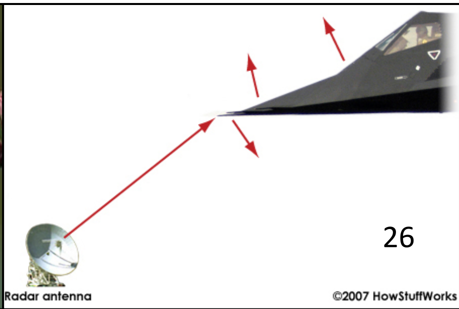
Why trajectories are important: Agility in Innovation

- Evolutionary versions of systems have characteristics that are different (for better or worse) than their "ancestors":
  - Ancestors may be earlier product models or biological species, but may also be earlier configurations of a current (reconfigurable) system instance, or earlier ideas in a sequence of design concepts for a single project system.

- Over multiple life cycles, systems evolve (or are selectively evolved) in response to their environment:
  - New opportunities
  - New threats

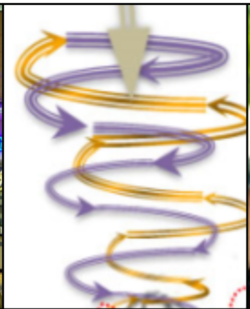
- The environment is itself made up of other evolving systems:
  - So, it would be more accurate to think of <u>co-evolution</u> of interacting systems (or evolution of the larger parent system)

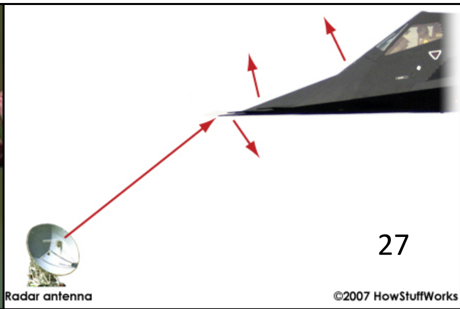




Why trajectories are important: Agility in Innovation

- Is a system of interest evolving rapidly and effectively enough in response to evolution of its:
  - Competitors?
  - Customers?
  - Prey?
  - Predators?
  - Opportunities?
  - Threats?
  - Resources?
- One definition of <u>Agile System</u> is a system that has that capability.
- Current example of great concern: Cyber security & internet of things

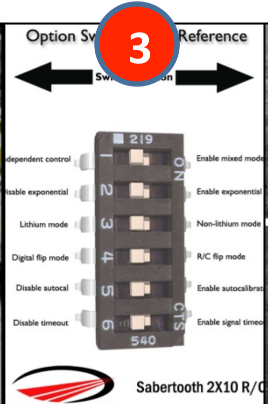


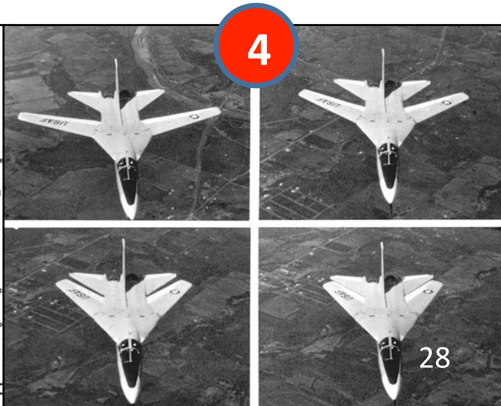


©2007 HowStuffWorks

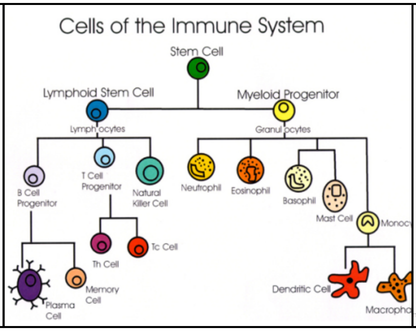
Why innovation trajectories are important: Agility in Innovation

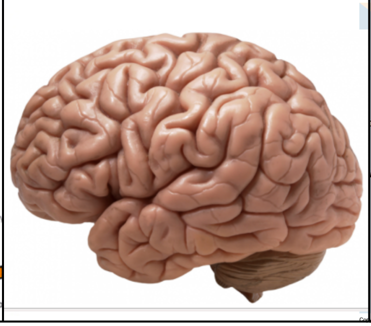
- As human-engineered systems become more mature, their ability to be re-configured advances to later in their life cycles:
  1. At first, all configuration occurs during design
  2. More advanced systems can be configured to order, at Manufacturing time (Dell pioneered; see also Ford pickup plant)
  3. Still more advanced systems can be configured after delivery, by their distributors, dealers, users, or maintainers.
  4. Even more advanced systems can reconfigure themselves while in operation.
- Biological scientists have referred to the “evolution of evolvability” as a major step in the early stages of living systems.

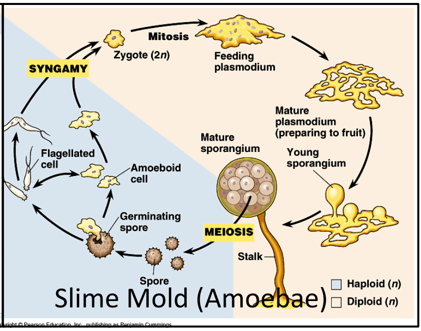




- Two advances increase the agility of in-service systems:
  - Composable architecture: flexibility through configurable architecture
  - Embedded information: (hardware/software combination; cyber-physical systems; increases flexibility)







Slime Mold (Amoebae)

- See INCOSE IW2015 MBSE Workshop, 01.24.2015:
  - Session on Modeling Agile Systems and Agile Modeling of Systems

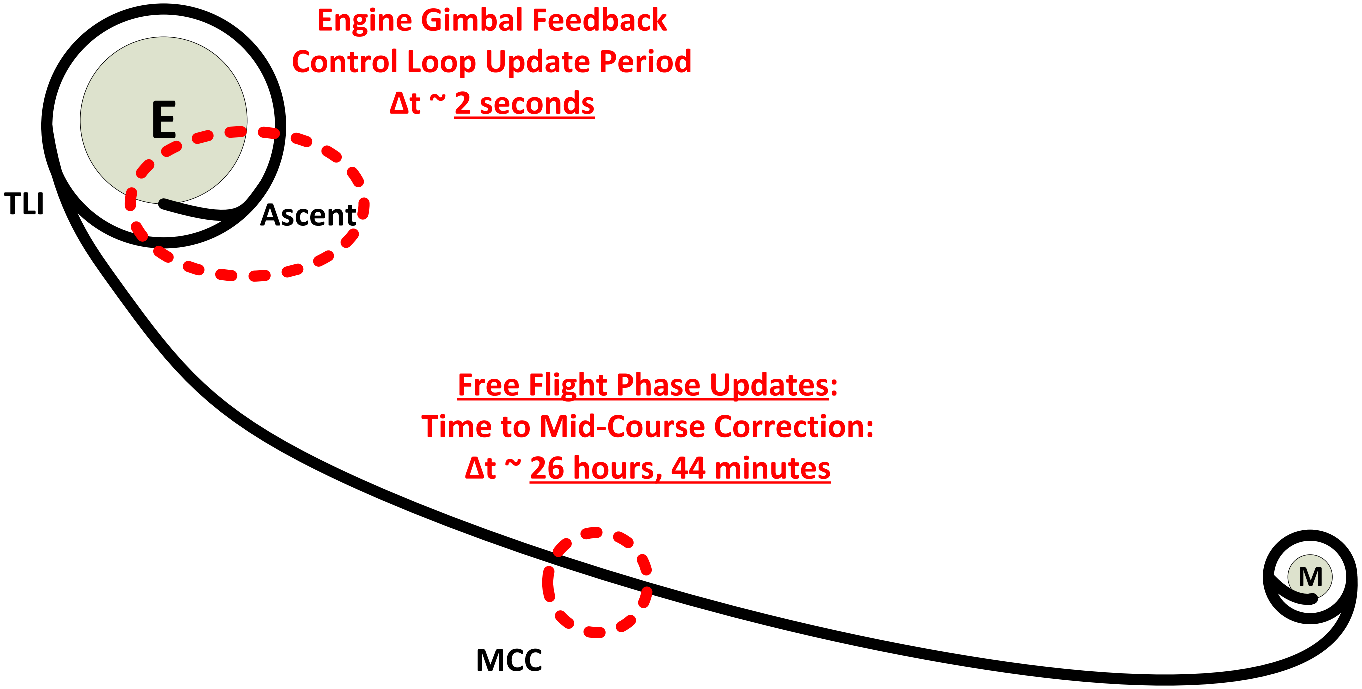
http://www.omgwiki.org/MBSE/doku.php?id=mbse:incose_mbse_iw_2015:breakout_out_session_agile_modeling

  - See INCOSE Agile Systems Engineering Life Cycle Model (ASELCM) Project, announced at IW2015:

http://www.incose.org/newsevents/announcements/Docs/AgileSELifeCycleModelProject-INCOSE-.pdf

Feedback & Correction Cycle Rate: A Hallmark of Agile Methods

An Apollo 11 Mission Question: Why was the Saturn V rocket engines' directional gimbals update cycle period throughout the Ascent Phase ~ 2 seconds, but the update cycle period of course direction during the Free Flight Phase was ~ 26 hours? [42,43]



Ascent Phase Updates:
Saturn V Launch Vehicle
Engine Gimbal Feedback
Control Loop Update Period
Δt ~ 2 seconds

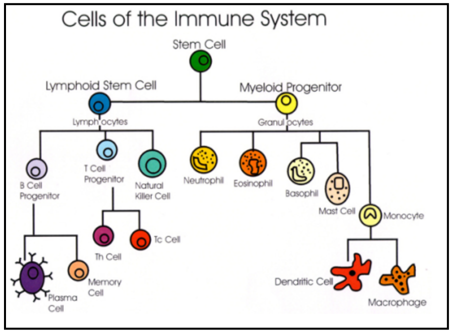
Free Flight Phase Updates:
Time to Mid-Course Correction:
Δt ~ 26 hours, 44 minutes

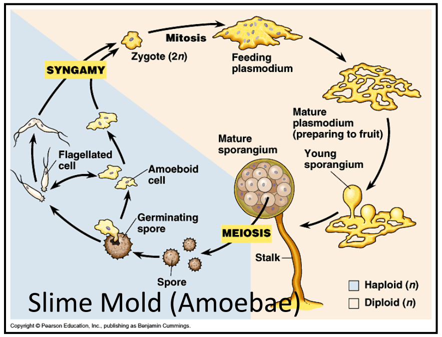
System Patterns Answer a Key Challenge to Agile Methods

- Another hallmark of agile methods is the repeated iterative release of a "complete enough" deliverables for some use to be made of them by the customer.
- For those considering use of agile methods, this often raises a key question / challenge:
  – How to produce a complete enough deliverable in each (time limited) sprint, for a complex system?
- Answer: Configured Patterns as draft deliverables—S*Patterns may be very quickly configured.

Accumulation of experience: Patterns as the DNA of systems

- Agile (fast adapting) systems take advantage of past experience:
  - An agile, composable system increases its agility if it “remembers what worked and did not”.
  - This implies learning from experience and retaining (remembering) those lessons
- Living systems invoke previously learned modes:
  - Immune systems retain memory of past antigen encounters and antibodies that worked.
  - Biological DNA retains memory of protein synthesis modes that apply under various stresses.
  - Brains retain memory of past situations and responses.

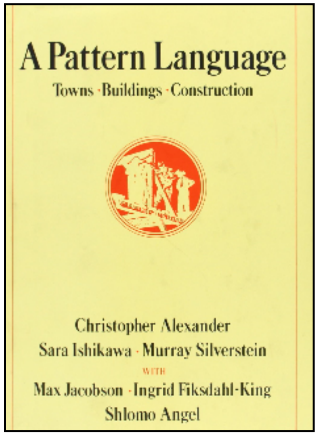




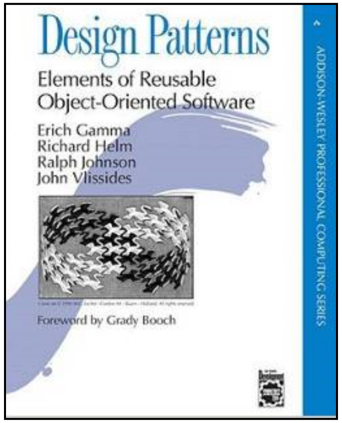
Slime Mold (Amoebae)

Accumulation of experience: Patterns as the DNA of systems

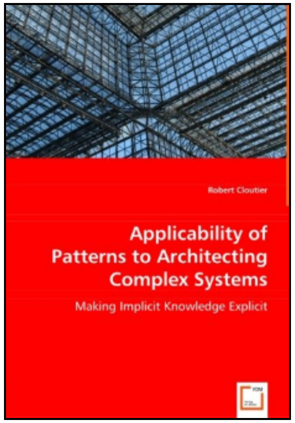
- Designers apply their accumulated human experience to future designs:
  - Informal writings, files, libraries, attempts at formal knowledge management
  - Pattern-based methods allow enterprises to more formally accumulate and reapply design patterns


A Pattern Language
Towns · Buildings · Construction
Christopher Alexander
Sara Ishikawa · Murray Silverstein
WITH
Max Jacobson · Ingrid Fiksdahl-King
Shlomo Angel

Civil Architecture


Design Patterns
Elements of Reusable Object-Oriented Software
Erich Gamma
Richard Helm
Ralph Johnson
John Vlissides
Foreword by Grady Booch
ADDISON-WESLEY PROFESSIONAL COMPUTING SERIES

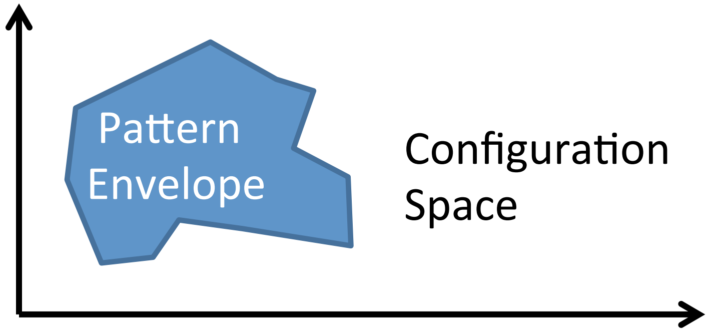
Software Design


Robert Cloutier
Applicability of Patterns to Architecting Complex Systems
Making Implicit Knowledge Explicit

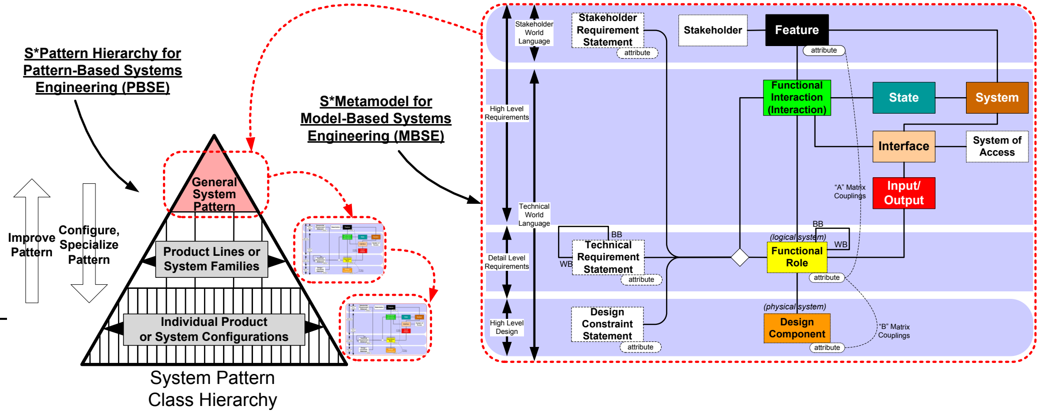
Systems

Accumulation of Experience: Patterns as the DNA of Systems

- Patterns describe regularities, across multiple instances:
  - Predict future from past
  - A the heart of the physical sciences
- Configuration space trajectories accumulate experience in patterns:
  - Increases the ability of (agile) systems to handle different situations.
  - As in configurable platforms, multi-mode systems, etc.
- Agile systems are more adaptable to different situations, but "mission envelopes" apply:
  - System "mission envelope" describes how widely a pattern applies.[25]
  - Adaptability, but may not anticipate refrigerators providing phone service!



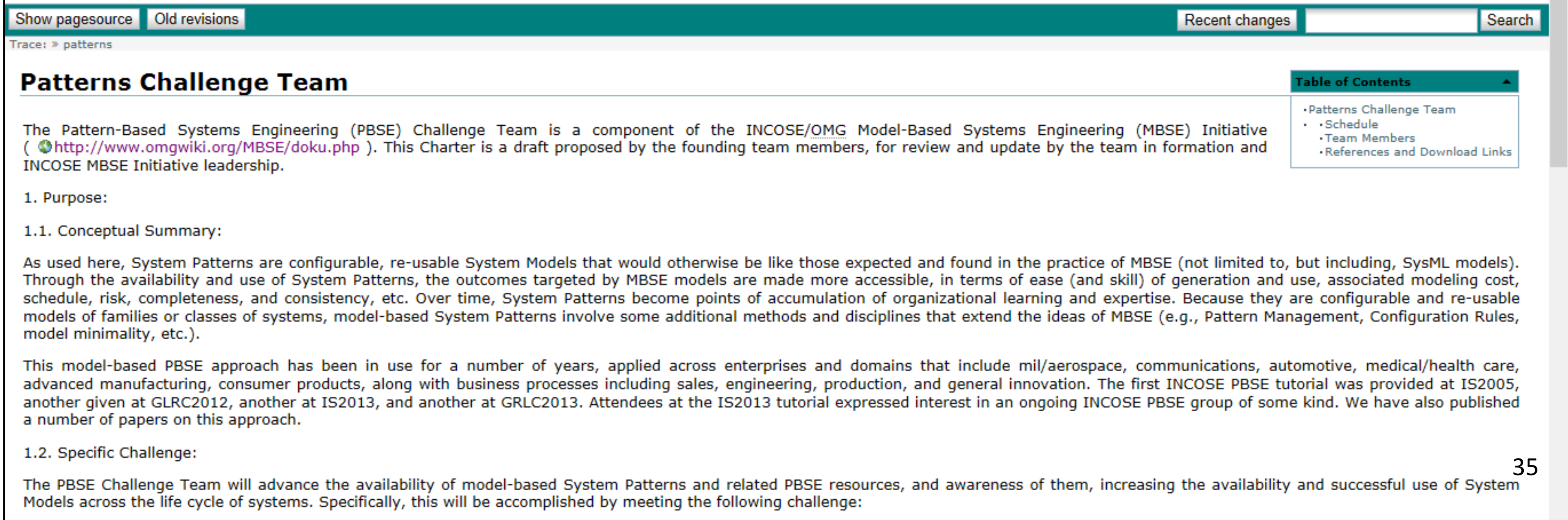
- The INCOSE/OMG MBSE Patterns Challenge Team is practicing the use of S*Patterns as demonstrations of the “smallest possible configurable model” of adaptable systems:
  - http://www.omgwiki.org/MBSE/doku.php?id=mbse:patterns:patterns


S*Pattern Hierarchy for Pattern-Based Systems Engineering (PBSE)
S*Metamodel for Model-Based Systems Engineering (MBSE)
Improve Pattern
Configure, Specialize Pattern
General System Pattern
Product Lines or System Families
Individual Product or System Configurations
System Pattern Class Hierarchy
Stakeholder World Language
High Level Requirements
Technical World Language
Detail Level Requirements
High Level Design
Stakeholder Requirement Statement
Stakeholder
Feature
attribute
Functional Interaction (Interaction)
State
System
Interface
System of Access
"A" Matrix Couplings
Input/ Output
BB
WB
Technical Requirement Statement
(logical system)
Functional Role
Design Constraint Statement
(physical system)
Design Component
"B" Matrix Couplings

WE SET THE STANDARD

MBSE WIKI

SE:PATTERNS:PATTERNS



Trace: » patterns

Patterns Challenge Team

The Pattern-Based Systems Engineering (PBSE) Challenge Team is a component of the INCOSE/OMG Model-Based Systems Engineering (MBSE) Initiative (http://www.omgwiki.org/MBSE/doku.php). This Charter is a draft proposed by the founding team members, for review and update by the team in formation and INCOSE MBSE Initiative leadership.

1. Purpose:

1.1. Conceptual Summary:

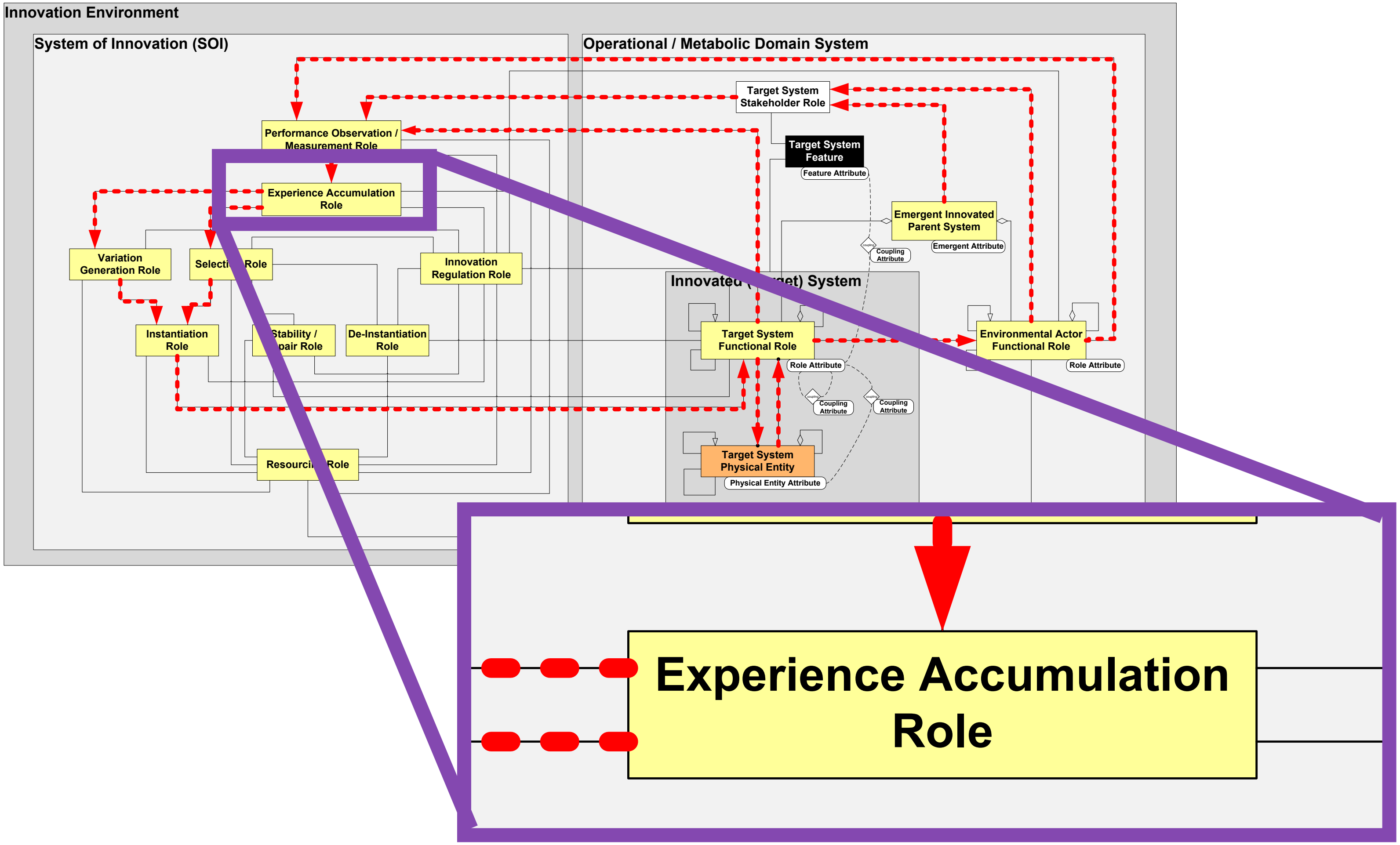
As used here, System Patterns are configurable, re-usable System Models that would otherwise be like those expected and found in the practice of MBSE (not limited to, but including, SysML models). Through the availability and use of System Patterns, the outcomes targeted by MBSE models are made more accessible, in terms of ease (and skill) of generation and use, associated modeling cost, schedule, risk, completeness, and consistency, etc. Over time, System Patterns become points of accumulation of organizational learning and expertise. Because they are configurable and re-usable models of families or classes of systems, model-based System Patterns involve some additional methods and disciplines that extend the ideas of MBSE (e.g., Pattern Management, Configuration Rules, model minimality, etc.).

This model-based PBSE approach has been in use for a number of years, applied across enterprises and domains that include mil/aerospace, communications, automotive, medical/health care, advanced manufacturing, consumer products, along with business processes including sales, engineering, production, and general innovation. The first INCOSE PBSE tutorial was provided at IS2005, another given at GLRC2012, another at IS2013, and another at GRLC2013. Attendees at the IS2013 tutorial expressed interest in an ongoing INCOSE PBSE group of some kind. We have also published a number of papers on this approach.

1.2. Specific Challenge:

The PBSE Challenge Team will advance the availability of model-based System Patterns and related PBSE resources, and awareness of them, increasing the availability and successful use of System Models across the life cycle of systems. Specifically, this will be accomplished by meeting the following challenge:

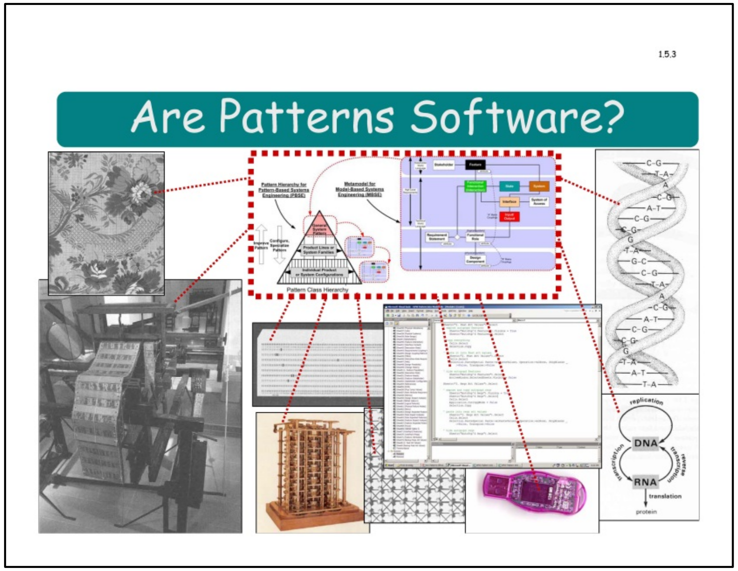
- INCOSE SSWG 2012: There is a universal model of innovation that includes all those stages; universal complex adaptive system: [30]


Innovation Environment
System of Innovation (SOI)
Operational / Metabolic Domain System
Target System Stakeholder Role
Performance Observation / Measurement Role
Target System Feature
Feature Attribute
Experience Accumulation Role
Emergent Innovated Parent System
Emergent Attribute
Coupling Attribute
Variation Generation Role
Selecti Role
Innovation Regulation Role
Innovated (get) System
Instantiation Role
Stability / pair Role
De-Instantiation Role
Target System Functional Role
Role Attribute
Environmental Actor Functional Role
Role Attribute
Coupling Attribute
Coupling Attribute
Target System Physical Entity
Physical Entity Attribute
Resourci Role
Experience Accumulation Role

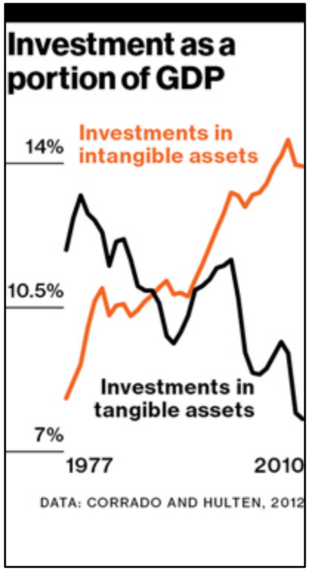
- This model already recognizes the key role of experience accumulation
- This model is being substantially updated by the INCOSE ASELCM Project [31]

Accumulation of Experience: Patterns as the DNA of Systems

- The accumulation of experience in systems suggests it is the future “software” of those systems:
  - Cyber-physical systems
- In the more literal sense of the current use of these terms, are patterns software?
  - A strong case can be made that S*Patterns already satisfy the contemporary definition(s) of (financially capitalizable!) “software” [16]


1.5.3

Are Patterns Software?



US$ crossover has already occurred!

Implications for Systems Engineering: Why S*Trajectories Are More than A Metaphor

- Just as modern geographic navigators have more powerful models, mathematics, and tools than their ancient counterparts,
  - So also may future agile systems project leaders have more powerful means of directional navigation throughout their projects.
- A related conjecture reported in the INCOSE SSWG Model Team report at IW2014:
  - *"S*Features (which describe fitness or value) define a vector field in S*Configuration Space, the equivalent of physical Potential, and the gradient of which is equivalent to physical Force on evolutionary configurations in this configuration space.*
  - *The path followed by an evolving system family moving on a path through configuration space, solely under the influence of Feature selection pressure, will satisfy the Principle of Stationary (or Least) Action."* [32,38]
- Among the powerful tools available to aid in this approach are:
  - Calculus of Variations and the Principle of Least Action [44]
  - Pontryagin Maximum Principle [45]
  - Theory of Optimization, Estimation, and Control, including Observability and Controllability [46]

Summary and Conclusions

1. Current procedure-focused systems engineering & innovation processes can be made more effective by increasing the focus on underlying information vs. procedure, with impacts:
   - Simplify, while Speeding and Improving Outcomes
   - Improved ability to understand and communicate current situation
   - More general Risk Management
   - Increased agility
2. There are very practical reasons to want to track the trajectory of system configurations, during development, during in-service life cycles, and across product line evolutions.
3. There is a minimal "genome" (S*Metamodel) that can provide a practical way to capture, record, and understand those trajectories, with significant business impact.
4. Patterns (configurable reusable models) can provide higher leverage means for implementing MBSE, tracking and exploiting system configuration trajectories, configured by selectable Stakeholder Features.
5. This has allowed us to create an MBSE-based version of 15288 Systems Engineering, using models and patterns, and to apply them in support of the Agile Systems Engineering Life Cycle Pattern.
6. Improved natural roles for automated aids, modelling tools, PLM systems:
   - e.g., gap views for (potentially agile) "steering", especially at Stakeholder Feature level
   - Realizing INCOSE Vision 2025

References

Representing Systems:

1. W. Schindel, “Requirements statements are transfer functions: An insight from model-based systems engineering”, *Proceedings of INCOSE 2005 International Symposium*, (2005).
2. ----------------, “What Is the Smallest Model of a System?”, *Proc. of the INCOSE 2011 International Symposium*, International Council on Systems Engineering (2011).
3. Long, David, “Model-Based Systems Engineering at the Age of Eight”, presentation at NDIA GVSETS Conf, Troy, MI, August, 2014.
4. Kaylan, Melik. “A World Without Maps”, The Wall Street Journal, 10.29-30.2013
5. Casagrande-Kim, Roberta, et al, NYU ISAW web site and bibliography on ancient cartography: http://isaw.nyu.edu/exhibitions/space/bibliography.html
6. Barkowsky, T, *Mental Representation and Processing of Geographic Knowledge*, Springer, Berlin, 2002.
7. Moerdijk, Ieke, “Descartes and the Geometrization of Algebra”, Descartes-Huygens Lecture, Radboud University, Nijmegen, The Netherlands, April 3, 2012.
8. Simmons, George F., *Introduction to Topology and Modern Analysis*, Chapter 10: Hilbert Spaces, McGraw-Hill, 1963.
9. “OMG Systems Modeling Language, Version 1.3”, Object Management Group, June, 2012.
10. W. Schindel, “Maps or Itineraries?: A Systems Engineering Insight from Ancient Navigators”, to appear in Proc. of INCOSE International Symposium 2015, July, 2015.
11. W. Schindel, “System Life Cycle Trajectories: Tracking Innovation Paths Using System DNA”, to appear in Proc. of INCOSE International Symposium 2015, July, 2015.

Patterns; Pattern-Based Systems Engineering:

12. Christopher Alexander, Sara Ishikawa, Murray Silverstein, Max Jacobson, Ingrid Fiksdahl-King, and Shlomo Angel. *A Pattern Language*. Oxford University Press, New York, 1977.
13. Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides. *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley Publishing Company, Reading, MA, 1995.
14. 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).
15. W. Schindel, “Pattern-Based Systems Engineering: An Extension of Model-Based SE”, INCOSE IS2005 Tutorial TIES 4, (2005).

16. -----------------, "Are Patterns Software?", ICTT System Sciences, January 2007.

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

18. 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).

19. 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

20. ----------------, "Introduction to Pattern-Based Systems Engineering (PBSE)", INCOSE Finger Lakes Chapter Webinar, April 26, 2012.

21. INCOSE/OMG Patterns Working Group 2013-14 http://www.omgwiki.org/MBSE/doku.php?id=mbse:patterns:patterns

22. Bill Schindel, Troy Peterson, "Introduction to Pattern-Based Systems Engineering (PBSE): Leveraging MBSE Techniques", in Proc. of INCOSE 2013 International Symposium, Tutorial, June, 2013.

23. ___________, "The Difference Between Whole-System Patterns and Component Patterns : Managing Platforms and Domain Systems Using PBSE", Proc. of Great Lakes Regional Conference on Systems Engineering, October, 2014.

<u>System Evolution and Innovation</u>:

24. Smolin, Lee, *The Life of the Cosmos*, Oxford, 1997.

25. Kauffman, Stuart, *The Origins of Order: Self-Organization and Selection in Evolution*, Oxford., 1993.

26. -----------------------, *Investigations*, Oxford, 2000.

27. Gould, Stephen Jay, *The Structure of Evolutionary Theory*, Harvard, 2002

28. W. Schindel, "Innovation as Emergence: Hybrid Agent Enablers for Evolutionary Competence" in Complex Adaptive Systems, Volume 1, Cihan H. Dagli, Editor in Chief, Elsevier, 2011

29. W. Schindel, S. Peffers, J. Hanson, J. Ahmed, W. Kline, "All Innovation is Innovation of Systems : An Integrated 3-D Model of Innovation Competencies ", Proc. of ASEE 2011 Conference, American Association for Engineering Education, (2011).

30. Schindel and Beihoff: "Systems of Innovation I: Models of Their Health and Pathologies", Proc. of INCOSE International Symposium, 2012.

31. W. Schindel, "Systems of Innovation II: The Emergence of Purpose", Proceedings of INCOSE 2013 International

32. INCOSE System Sciences Working Group, Systems of Innovation Project web site: https://sites.google.com/site/syssciwg/projects/o-systems-of-innovation

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

34. Rick Dove, "Agile Systems and Processes—Driving Architecture with ConOps and Response Situation Analysis (Agile 102)", Paradigm Shift, International, September 16, 2013.

35. Rick Dove, "Security R Us: Systems Engineering is the High Ground", INCOSE Biomedical Healthcare Working Group, April 24, 2014.

36. Rick Dove, Bill Schindel, "Modeling Agile Systems and Agile Modeling of Systems", to appear in : INCOSE IW2015 MBSE Workshop, 01.24.15.

37. Ashlee Vance, "Updates Available", article in "Technology" section of *Bloomberg Business Week*, Sept 8-14, 2014, pp. 30-32.

38. Gary Smith, Tom Marzolf, Bill Schindel, "Report of the SSWG SP/SP Modeling Sub-Team", INCOSE IW2014, Los Angeles, CA, Jan. 27, 2014.

Other References:

39. ISO/IEC 15288: Systems Engineering—System Life Cycle Processes. International Standards Organization (2014).

40. INCOSE Systems Engineering Handbook: A Guide for System Life Cycle Processes and Activities, Version 4, International Council on Systems Engineering (2014).

41. "A World in Motion: Systems Engineering Vision 2025", INCOSE, 2014.

42. "Instrument Unit Fact Sheet: Saturn V News Reference", IBM Federal Systems Division, 1968.

43. W. David Woods, Kenneth D. MacTaggart and Frank O'Brien, "Apollo 11 Flight Journal: Day 2, Part 1: Mid Course Correction", http://history.nasa.gov/ap11fj/05day2-mcc.htm , updated 2009.

44. Levi, Mark, *Classical Mechanics with Calculus of Variations and Optimal Control*, American Mathematical Society, Providence, Rhode Island, 2014.

45. Pontryagin, L. S., Boltyanskii, V. G., Gamkrelidze, R. V., and Mishchenko, E. F., The Mathematical Theory of Optimal Processes, transl. by D. E. Brown, Macmillan, New York, 1964.

46. Bryson, A. E., Ho, Y. C., Applied Optimal Control: Optimization, Estimation, and Control, Taylor & Francis Publishers, 1975.

Speaker

Bill Schindel is president of ICTT System Sciences (www.ictt.com), a systems engineering company.

His 40-year engineering career began in mil/aero systems with IBM Federal Systems, Owego, NY, included service as a faculty member of Rose-Hulman Institute of Technology, and founding of three commercial systems-based enterprises.

He has led and consulted on improvement of engineering processes within automotive, medical/health care, manufacturing, telecommunications, aerospace, and consumer products businesses.

Schindel earned the BS and MS in Mathematics. At the 2005 INCOSE International Symposium, he was recognized as the author of the outstanding paper on Modeling and Tools. Bill co-led a 2012-14 research project on the science of Systems of Innovation within the INCOSE System Science Working Group, and currently co-leads the Patterns Challenge Team of the OMG/INCOSE MBSE Initiative. He is also a co-lead of the INCOSE Agile Systems Engineering Life Cycle Model (ASELCM) Project.

Bill is an INCOSE CSEP, and president of the Crossroads of America INCOSE chapter.