25th anniversary annual INCOSE international workshop
Los Angeles, CA
January 24 - 27, 2015

ATTACHMENT 1:
LIMITED SAMPLE FROM DRAFT SESSION SLIDES

The Agile Systems Pattern

An MBSE-Based System Pattern, with Implications for Agile Modeling

Bill Schindel, ICTT System Sciences
schindel@ictt.com

Objectives of Overall Breakout Session on Agile Modeling and Modeling Agile Systems

- Gain a broader understanding of Agile Systems-engineering and the Engineering of Agile-Systems, and the common fundamental Agile Architecture Pattern of both that enables effective response in uncertain, unpredictable, and evolving SE and operational environments.
- Learn how formal model-based System Patterns can expand the fundamental Agile Architecture Pattern with necessary agile-enabling details for fleshing out an agile SE process and agile system design.
- Understand the role and impact of accumulated system patterns within Agile Systems.
- Learn about Pattern-Based Systems Engineering (PBSE), and how Agile Modeling is facilitated by PBSE.
- Learn how S*Patterns express model-based system patterns.
- Find out about the 2015/16 traveling workshop Agile System Engineering Life Cycle Model (ASELCM) Fundamentals project, to occur in the US and UK, along with how and why you and your organization might want to participate

Assumption: Some awareness of the general ideas of Agile Systems from this breakout's preceding sub-session, or the listed References

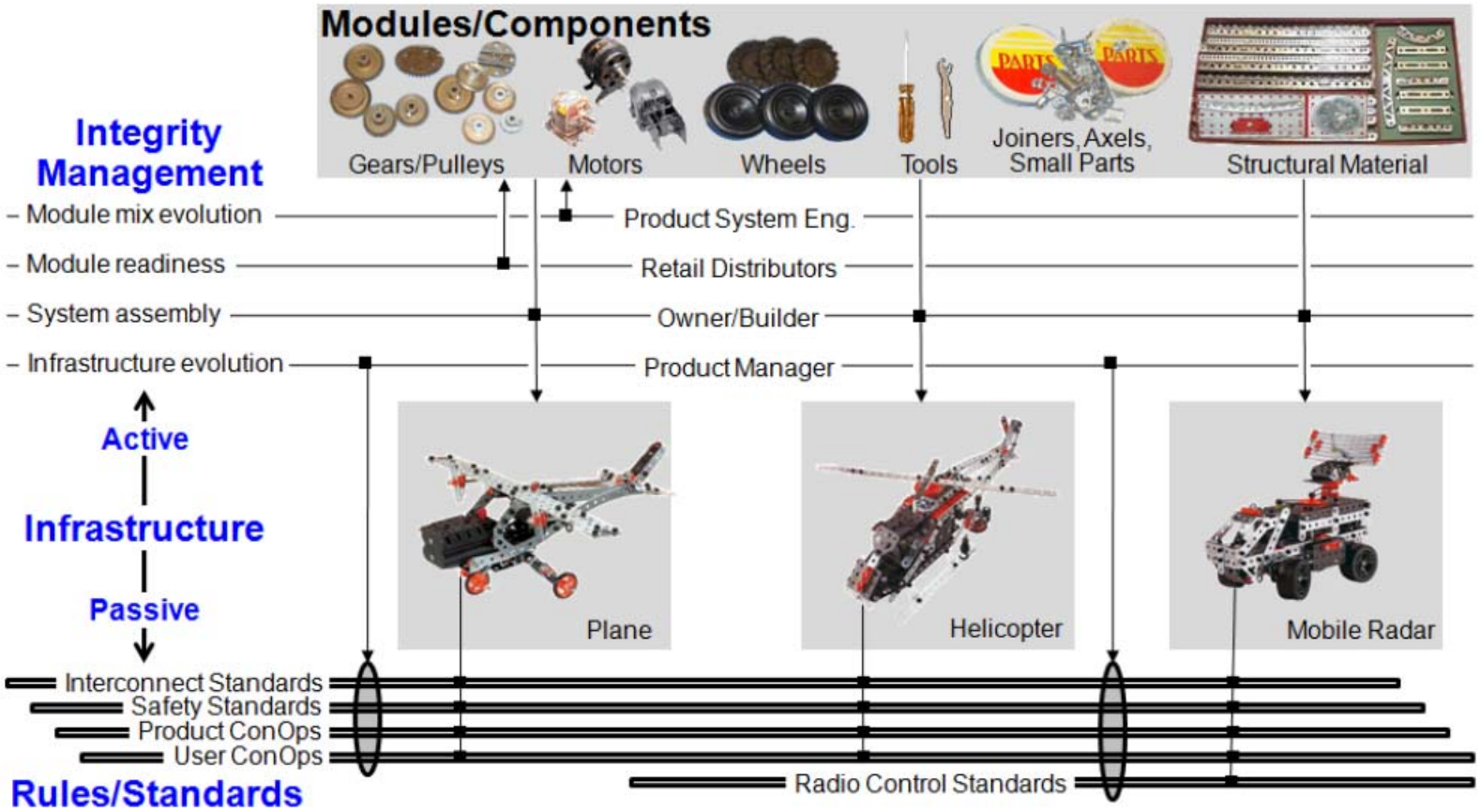
Goals of this sub-session

- Review a summary of major segments of the Agile Systems Pattern.
- From this example, learn about model-based, reusable, configurable representations of system patterns using the S*Metamodel.
- Understand the implications of the "Experience Accumulation" subsystem of the Agile Systems Pattern, for Agile Modeling, as well as other implications.
- Find out how to learn more.

A Peek Ahead

The Agile System Pattern will capture (in S*Model) the key ideas associated with the pre-MBSE Agile System Architecture:

- As in (Dove and LaBarge, 2014)



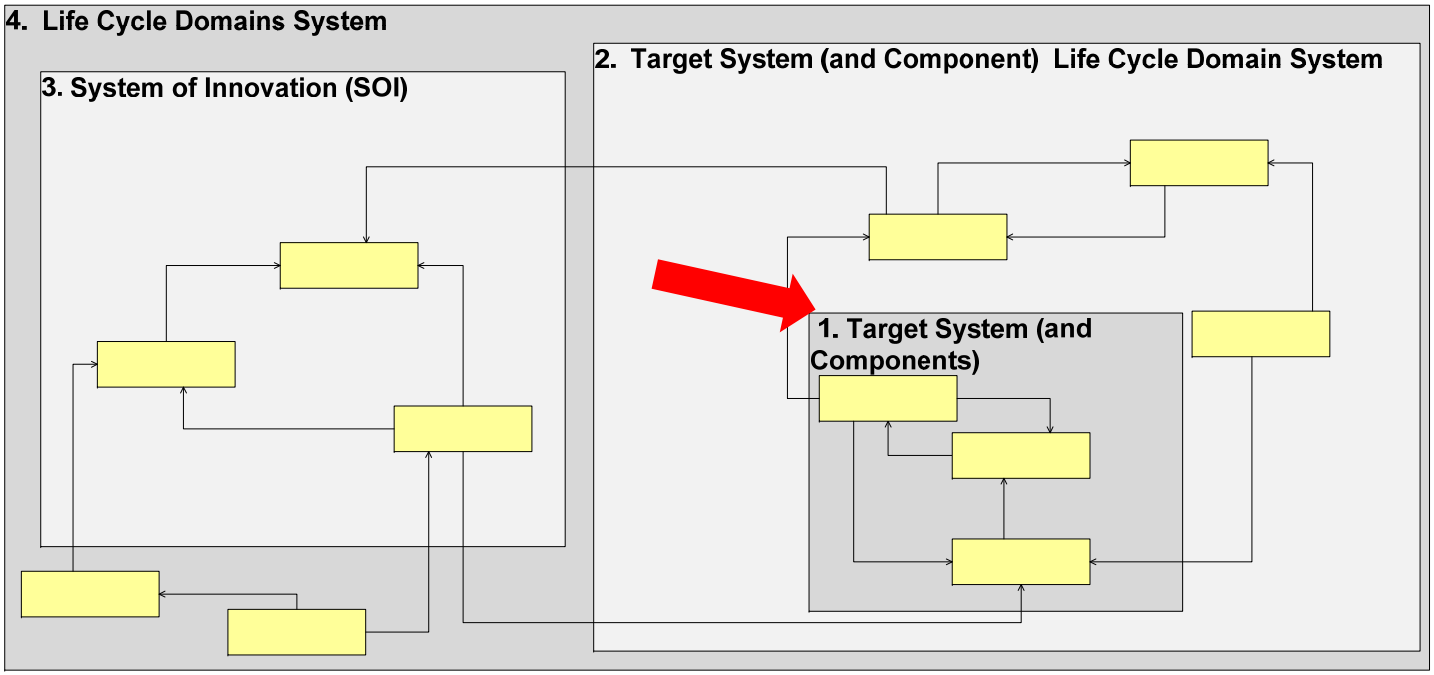
Agile System Pattern Content

- Summary (See Attachment for SysML version):
  - Domain Model
  - Logical Architecture Model
  - Physical Architecture Model
  - Input-Outputs, Interfaces, Systems of Access
  - Features Model
  - Attribute Coupling Model
  - Interactions Model
  - State Model
  - Requirements Model

The Agile System Domain Model

System 1: The **Target System (and Components):** (Definition) The logical system of interest, which results from, or is subject to, innovation.

- Its behavior, characteristics, or performance are targets of the innovation (change, adaptation) process we'll introduce later.
- It is potentially agile.
- Examples include aircraft, satellites, the human immune system, restaurants, birds, and the health care delivery system.

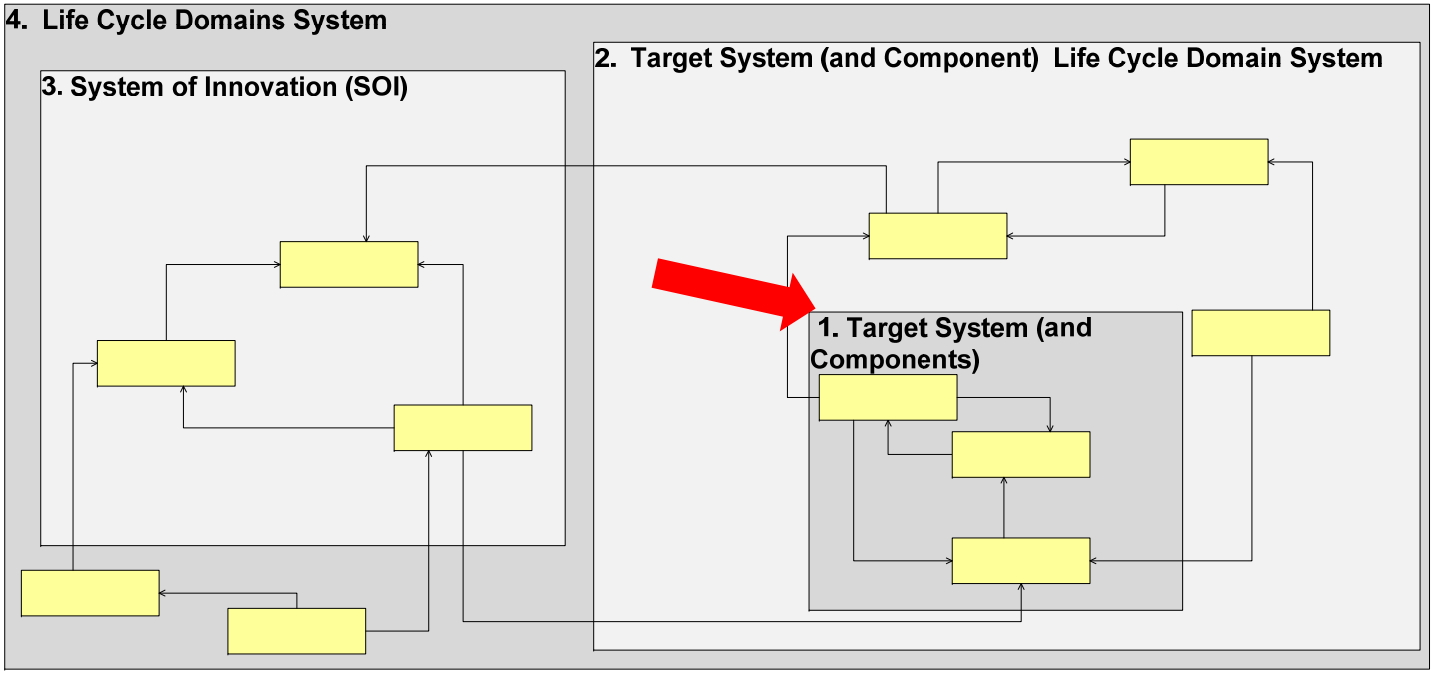


Internal component roles shown (yellow shapes) are notional, and will be identified and defined later.

The Agile System Domain Model

System 1: The **Target System (and Components):** (Definition) The logical system of interest, which results from, or is subject to, innovation.

- The Components maintained for integration into a Target System, but not yet integrated, are included in this domain.
- Notice that this idea can apply at multiple additional levels (e.g., SOS, System, Component, etc.)

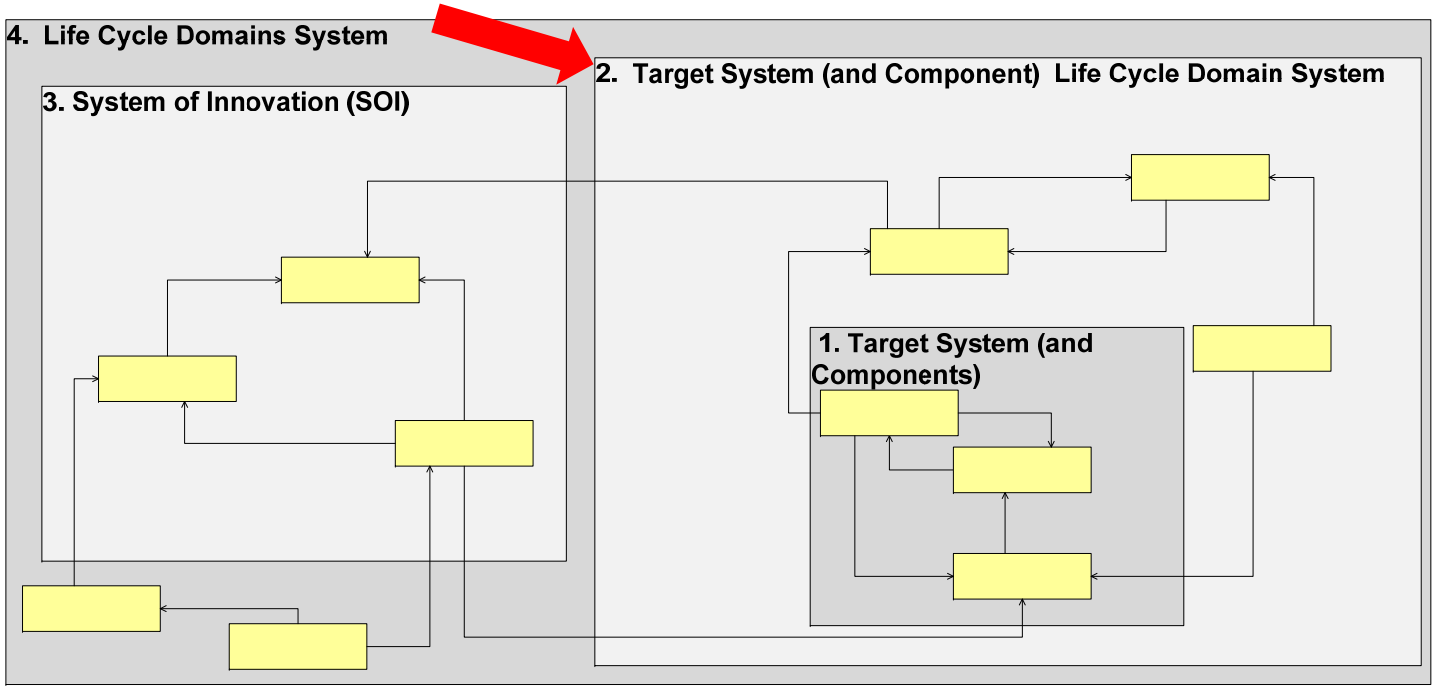


Internal component roles shown (yellow shapes) are notional, and will be identified and defined later.

The Agile System Domain Model

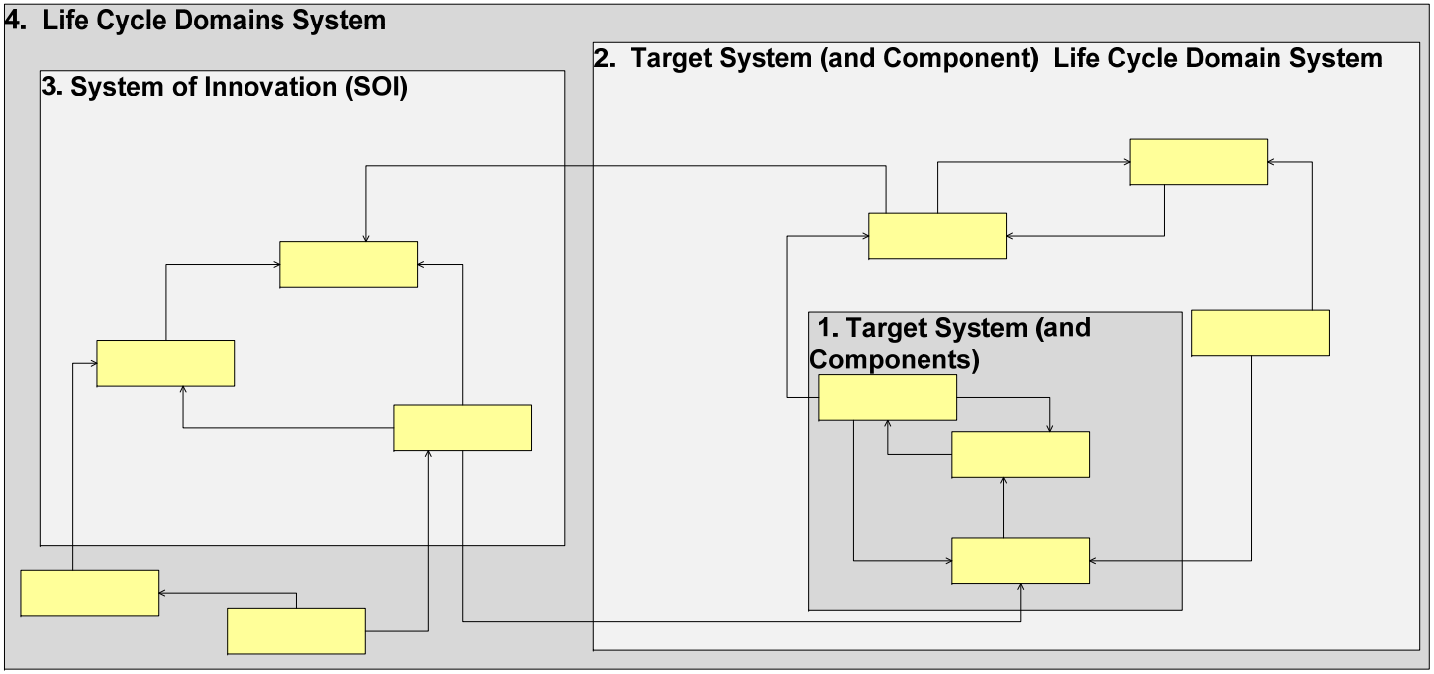
System 2: The **Target System (and Component) Life Cycle Domain System:** (Definition) The logical system within which the Target System will exist during its life cycle, when "in service" or otherwise. This domain includes all actors with which the Target System will directly interact during its life cycle:

- This includes any system that directly manages the life cycle of an instance of a Target System (or a Component)—production and integration systems, maintenance and operations systems, and others.



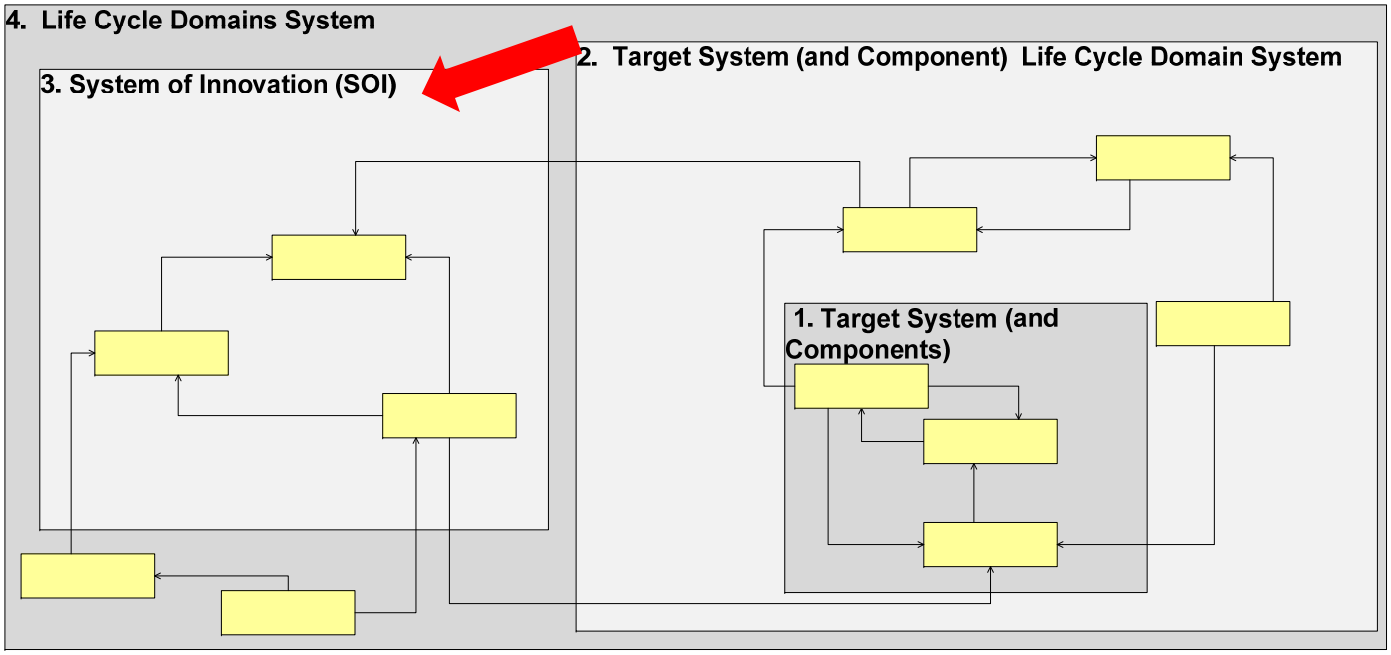
Internal component roles shown (yellow shapes) are notional, and will be identified and defined later.

- Again, remember that these are logical (behavioral) roles. In realized physical systems, a single physical system may behave as both a Target System and a system that produces, modifies, reconfigures, or otherwise manages a Target System, by having roles from each allocated to it.
- For purposes of this logical roles description, they have been identified separately.
- We will add the physical components to the model shortly.



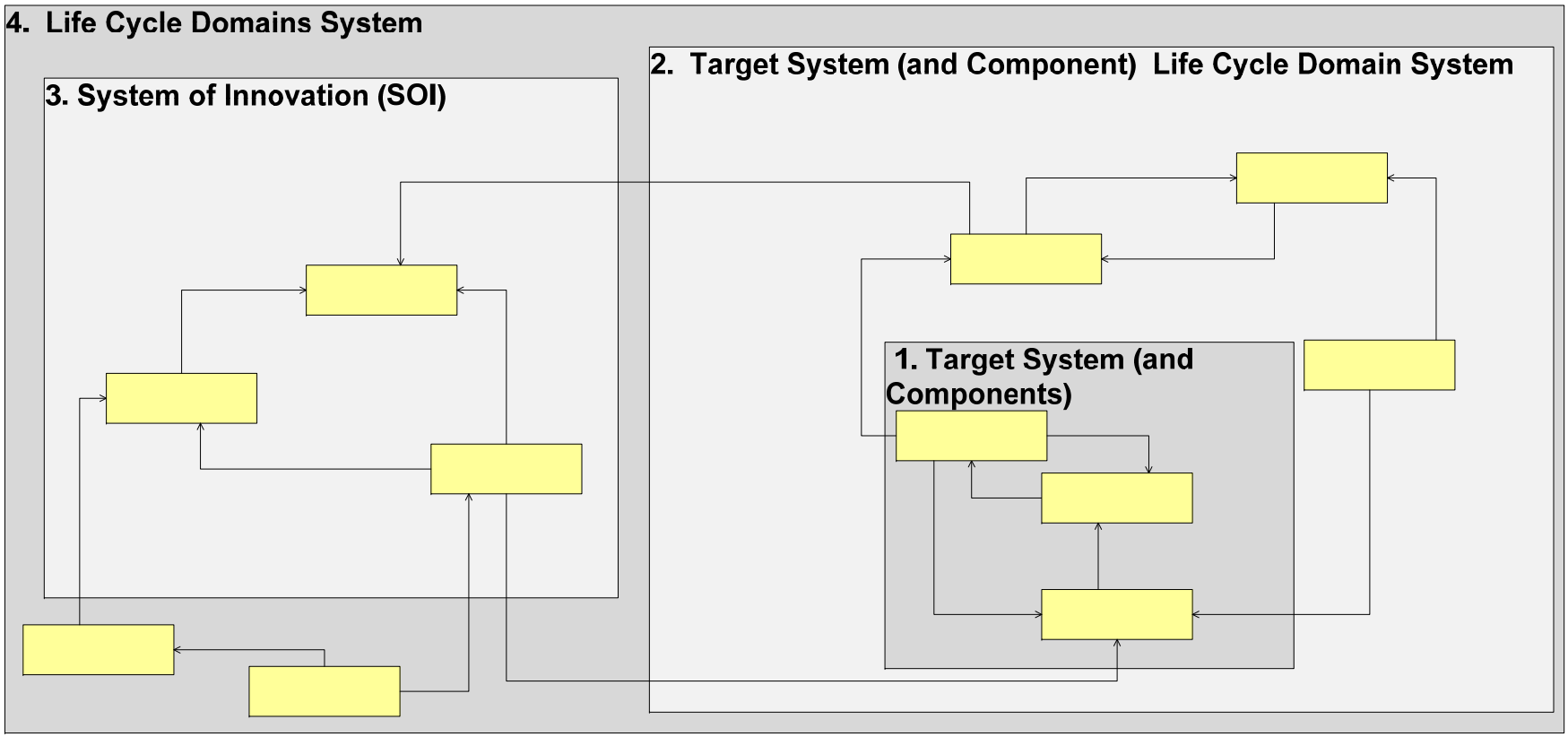
System 3: The **System of Innovation:** The logical system responsible for creating the possibility of (not production of) instances of Target System(s) with new or modified capabilities:

- Includes distillation of new knowledge (by observation) about Target Systems, their life cycle management, and their environmental domains, for future use.
- Also includes creation of instances of new production or other life cycle management capabilities for Target Systems, but not new instances of Target Systems.
- Engineers might think of this as the Engineering Process or the Development Process, but we have given it a more general name--to remind us that an innovation "competitor" may be operating from a cave or kitchen table, lacking a "recognized" engineering process; or, it might be a biological process that did not attend engineering school; or it might be some other type of innovation process, which we will study here.



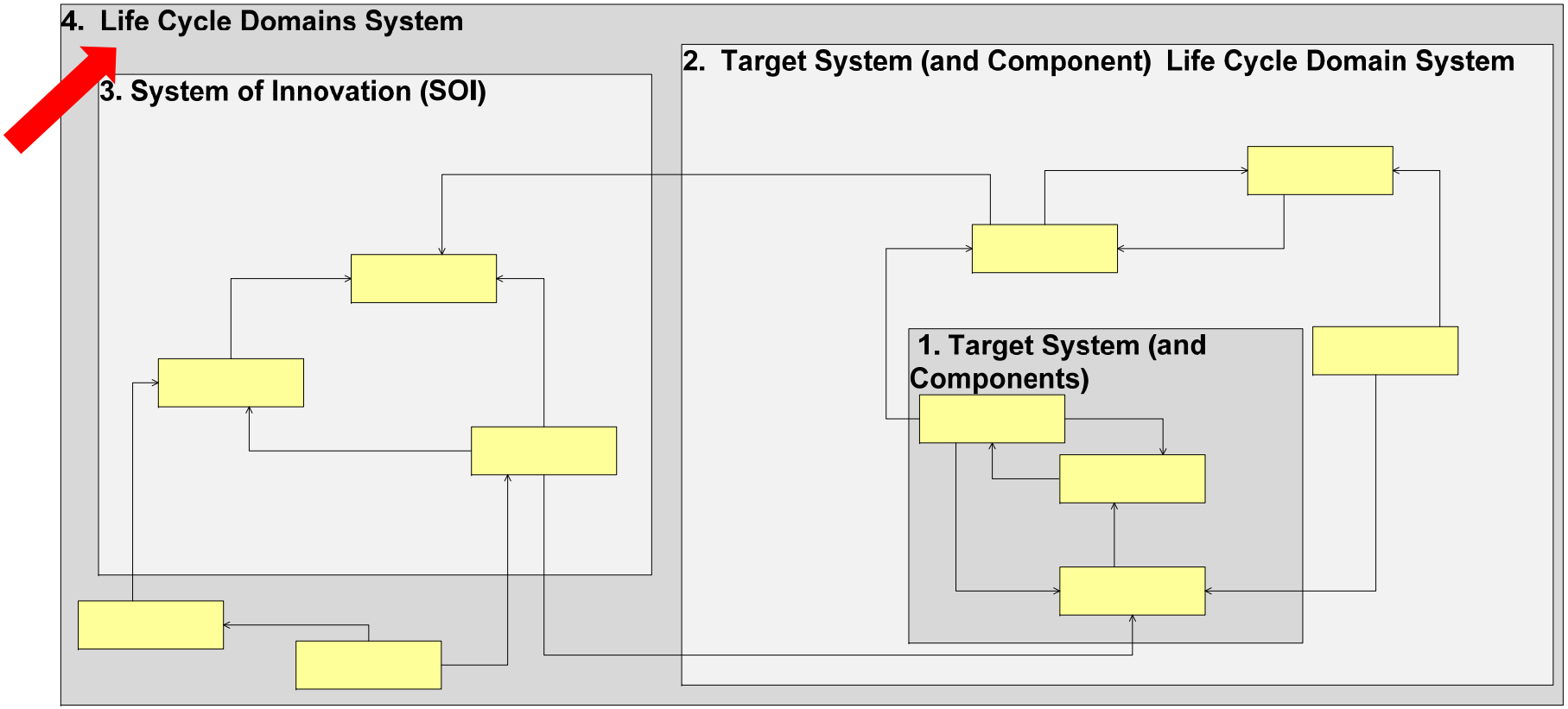
Internal component roles shown (yellow shapes) are notional, and will be identified and defined later.

- Summary so far:
  - System 2, the Target System Life Cycle Domain System produces and modifies instances of System 1, the Target Systems (and Components).
  - System 3, the System of Innovation, produces new abilities to do so, including knowledge.



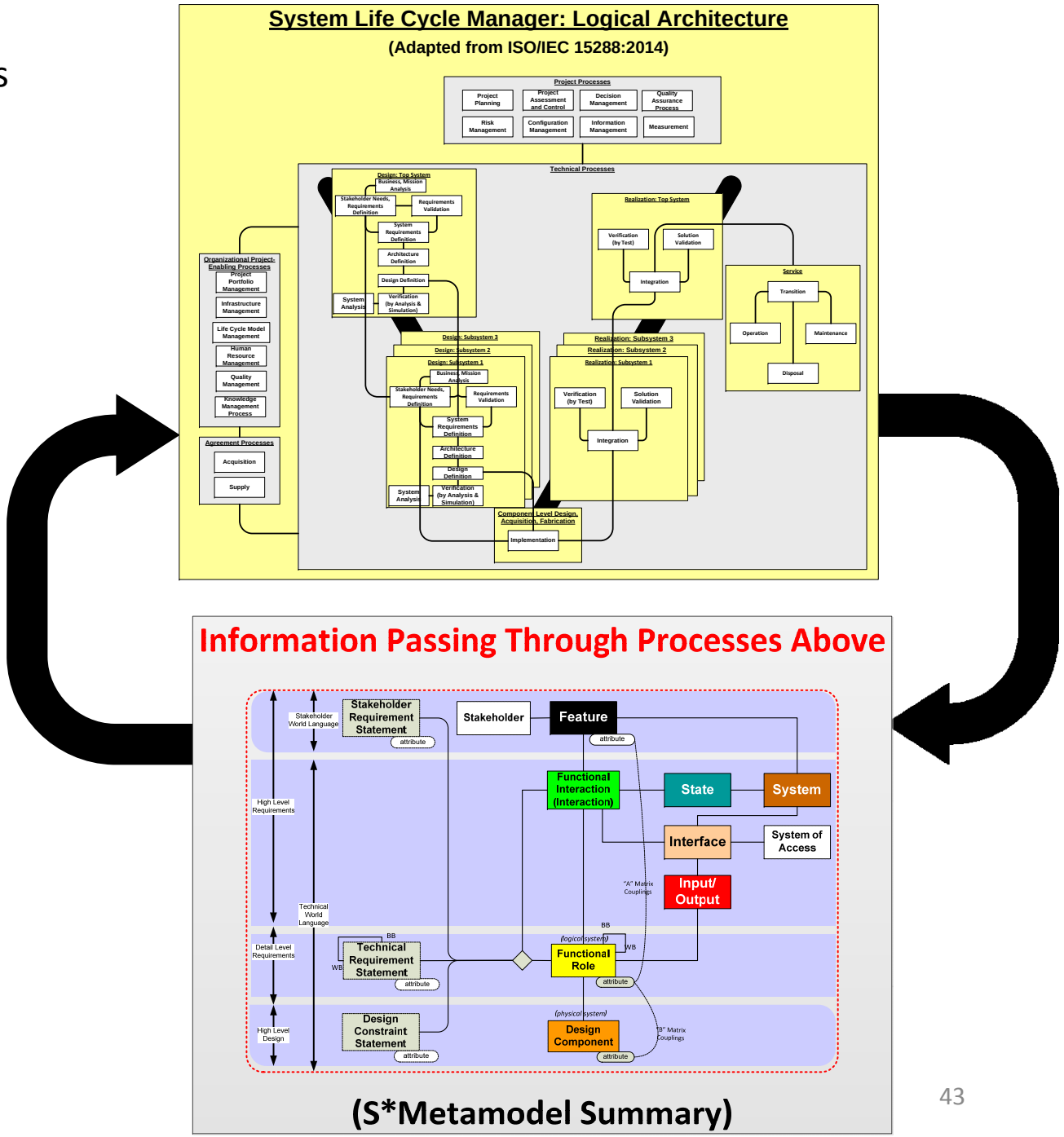
The Agile System Domain Model

System 4: The **Life Cycle Domains System**, consisting of the entire environment of the Target System, along with that Target System, across all of its life cycle stages, including innovation:



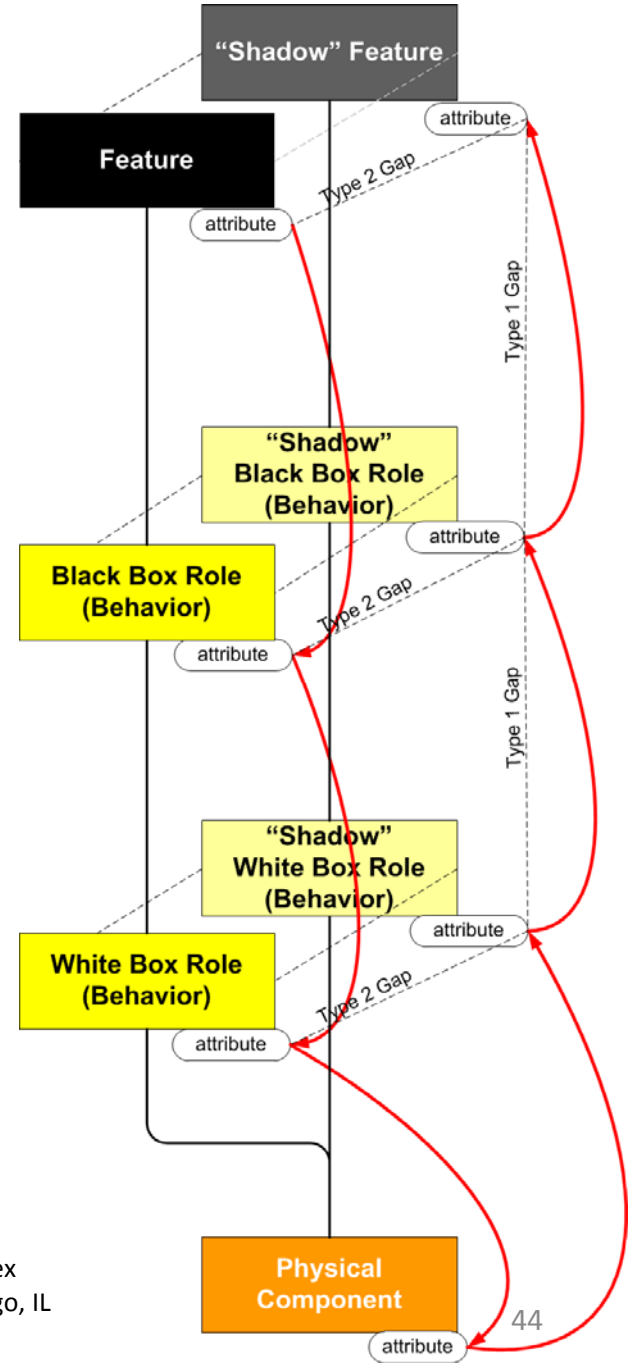
Process vs. Information:

- The S*Metamodel describes the MBSE information that passes through life cycle processes, as S*Models.
- Using PBSE accelerates this process, by basing S*Model information on knowledge-managed S*Patterns.
- None of this requires any specific sequence or order of processes, which may be concurrent or otherwise, depending on strategy.
- What is the “agile trajectory” through S*Space?
- Agile Scrum strategy is to “sprint” short distances in S*Space, with fixed time and resource budgets.
- How are the “trajectory deltas” planned for each Agile Scrum?



Agile Trajectories Are Driven By Gradient Field & Gaps:

- Type 1 and Type2 Gaps measure the gradient field used to establish incremental Agile Scrum “sprint” deltas.
- Driven by Stakeholder Feature Attribute Coupling metrics



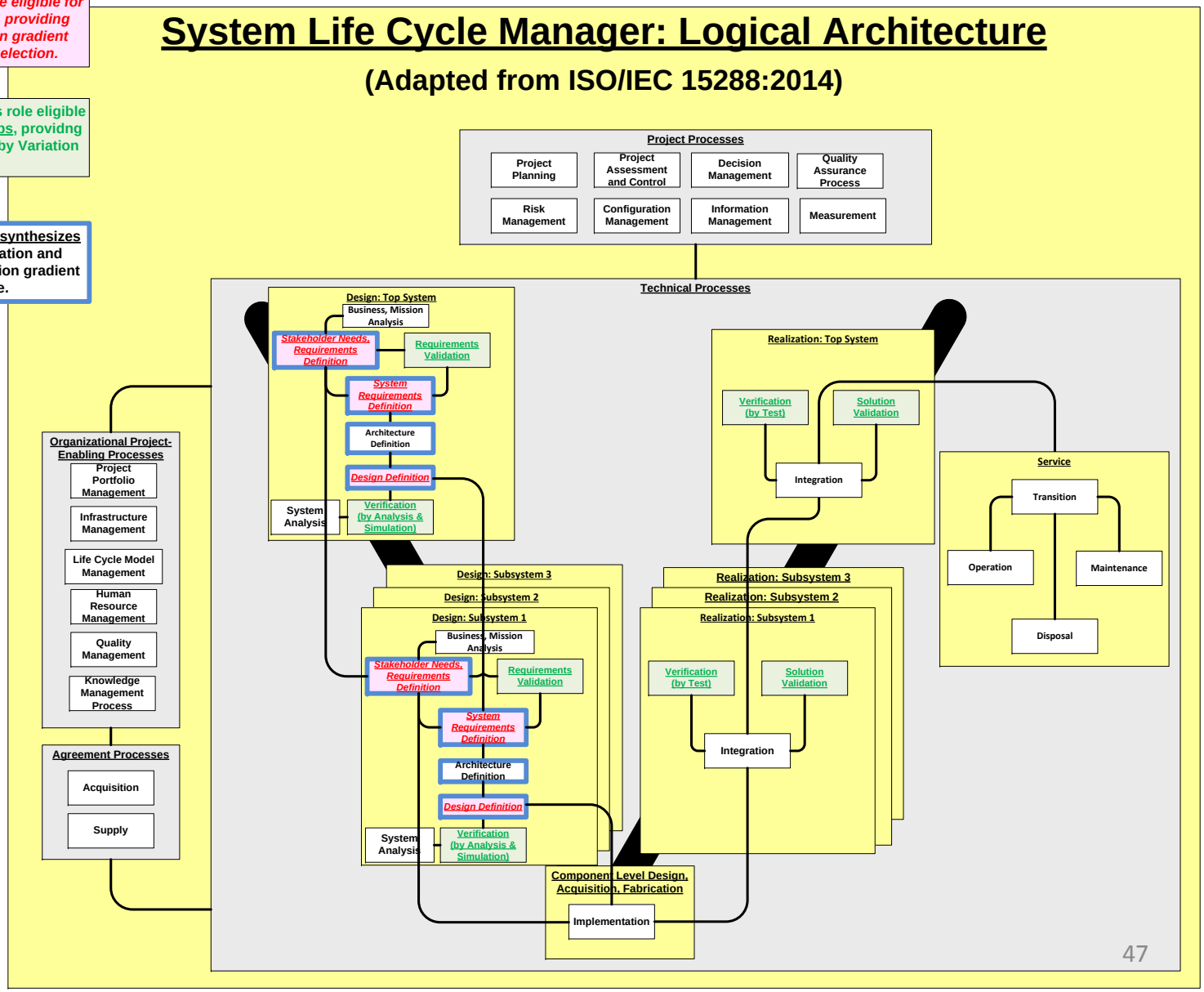
From: “Innovation as Emergence: Hybrid Agent Enablers for Evolutionary Competence”, W. Schindel, Complex Adaptive Systems, Volume 1, Cihan H. Dagli, Editor in Chief, *Procedia Computer Science*, Elsevia, 2011, Chicago, IL

Methodology for Selecting Agile Scrum Delta Vectors

Red Italics Underline means role eligible for measurement of Type 1 gaps, providing feedback to setting innovation gradient direction by Variation and Selection.

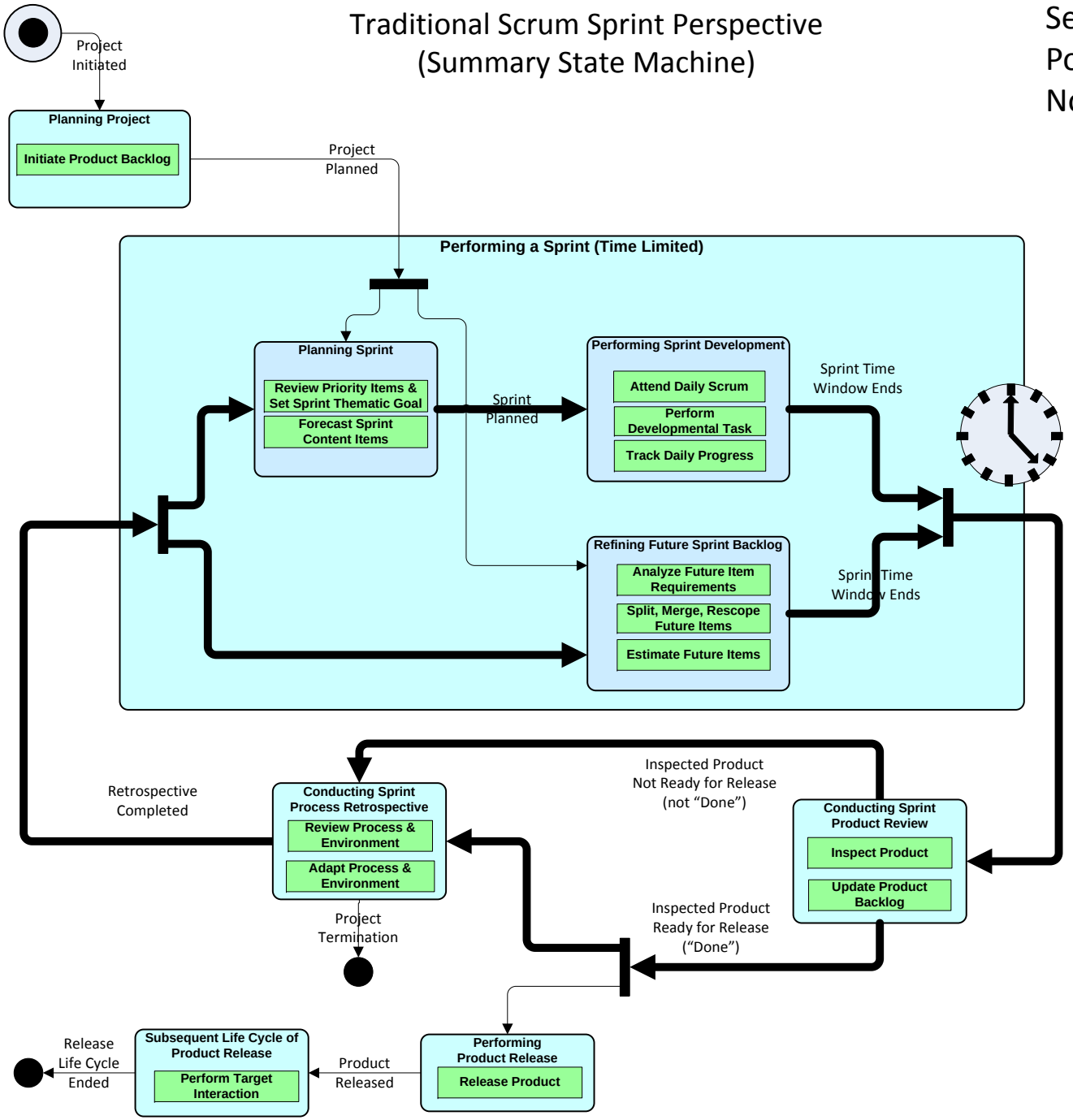
Green Roman Underline means role eligible for measurement of Type 2 gaps, providng innovation gradient direction by Variation and Selection

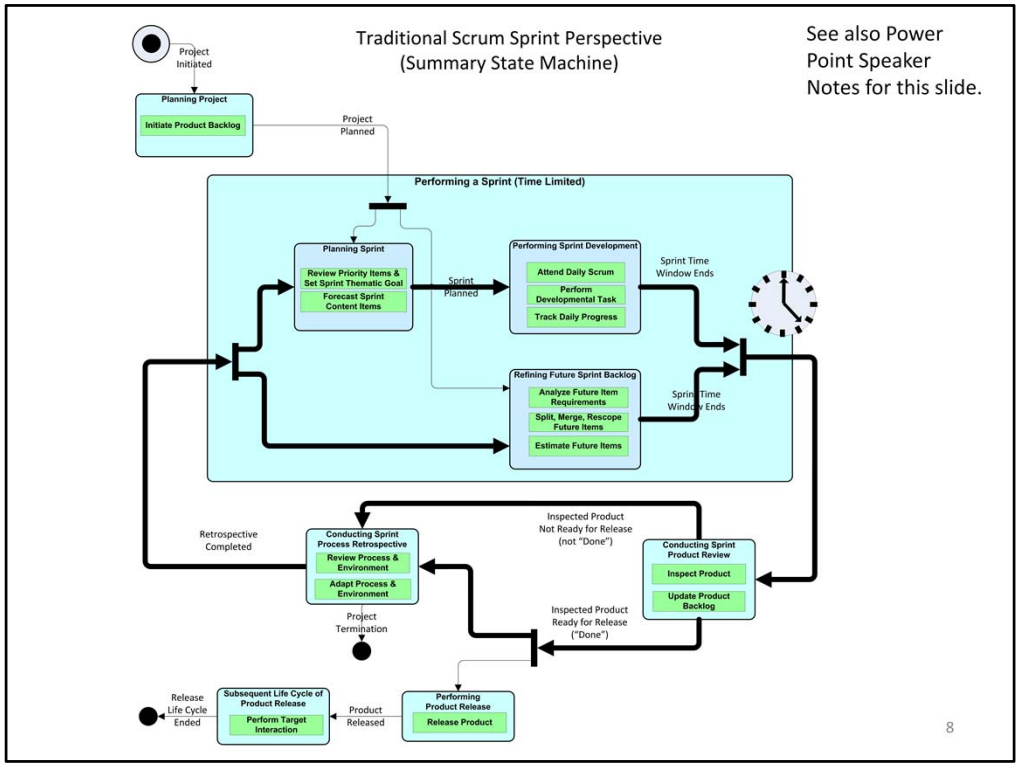
Thick blue outline means role synthesizes information eligible for Variation and Selection, and setting innovation gradient direction in S*Space.



Traditional Scrum Sprint Perspective (Summary State Machine)

See also Power Point Speaker Notes for this slide.

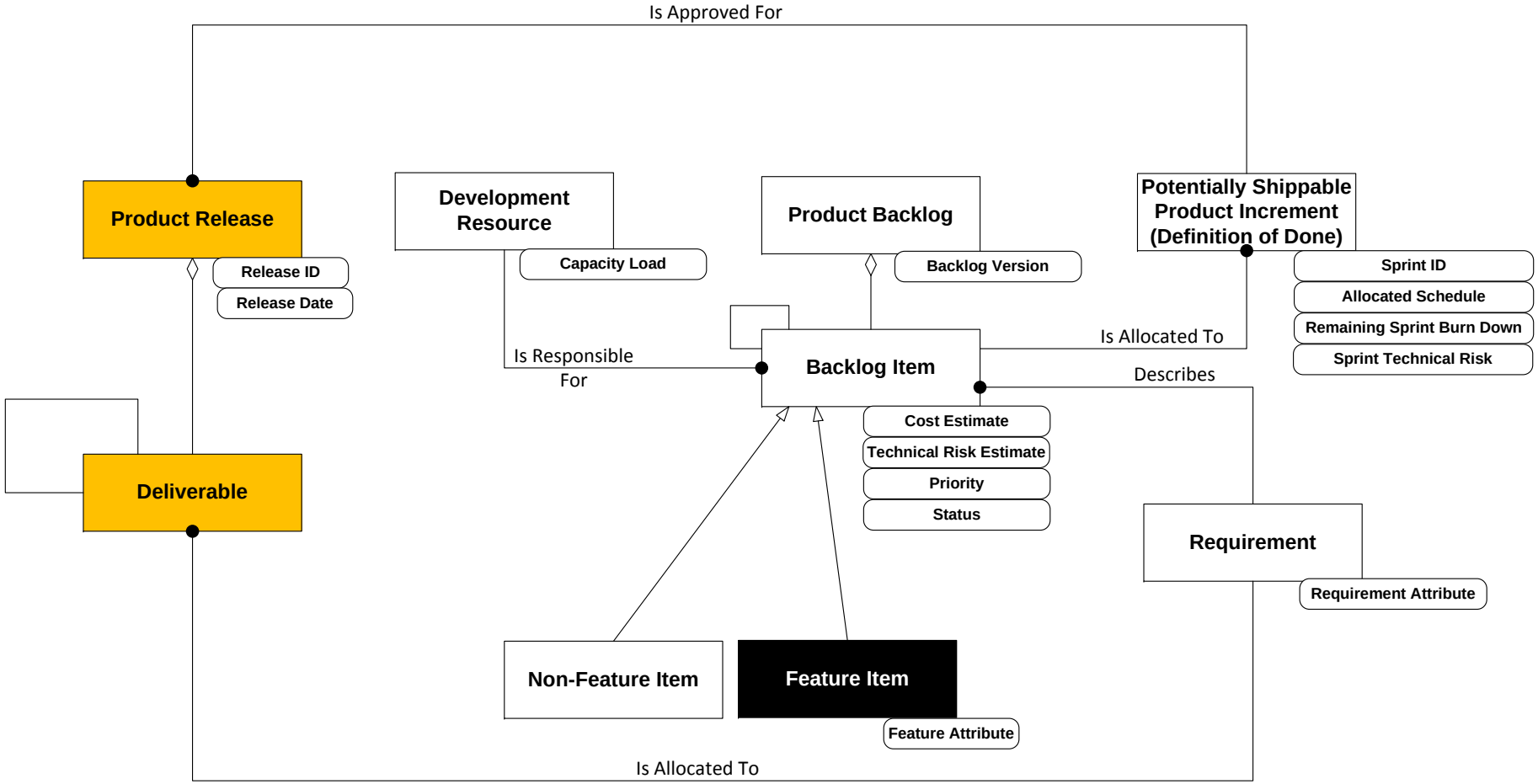


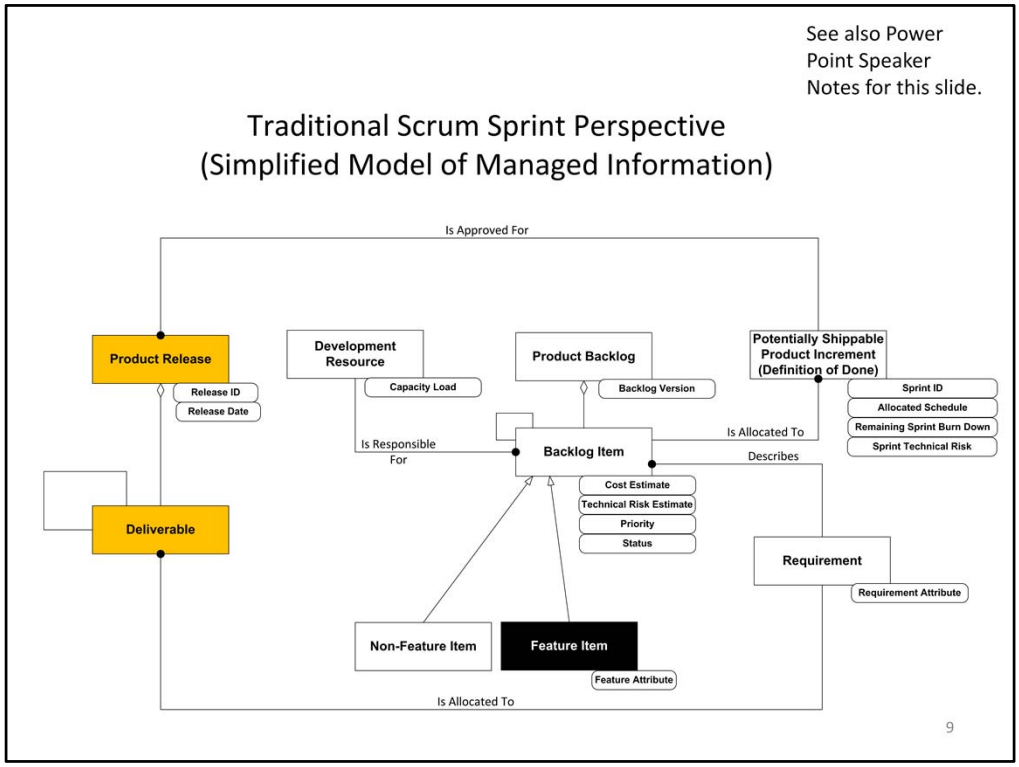


- This diagram is one way to represent traditional agile development, emphasizing in this view the “states” that the process passes through, and the “interactions” that occur during those states.
- The leading feature of agile development is that it is performed in a series of (short) defined time increments, instead of defined content increments.
- These time increments are called “sprints”, in recognition of their short (not more than 30 days) length.
- Content of these increments is nevertheless still prioritized and planned (important!), but in actual execution it is the elapse of time which signals end of a sprint, and not the completion of the planned work.
- Planned deliverable content of each sprint is chosen so that, if achieved, it would be complete enough to incrementally deliver to the customer—including sufficiently tested, documented, etc. , even though not yet complete as a whole project yet.
- After each sprint, inspection of the incremental delivered product in (at least simulated) “operational use”, along with internal observation of the development process, are used to generate feedback for subsequent sprints.
- The agile process is thus inherently small step exploratory, instead of a top-down large commitment “cliff”—a strategy recognizing inherently unfamiliar, unknown, dynamically changing, or otherwise risky situations.

See also Power Point Speaker Notes for this slide.

Traditional Scrum Sprint Perspective (Simplified Model of Managed Information)

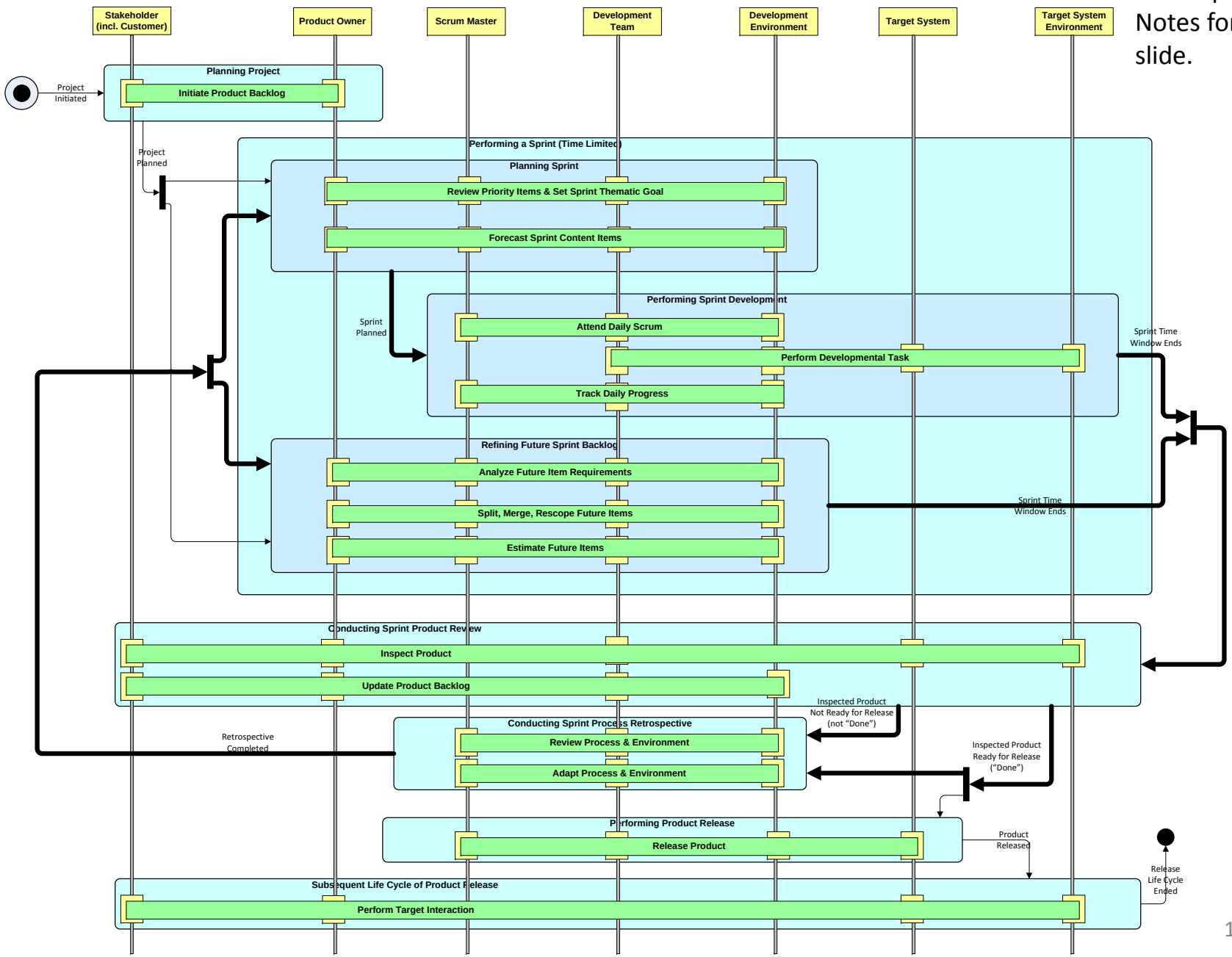


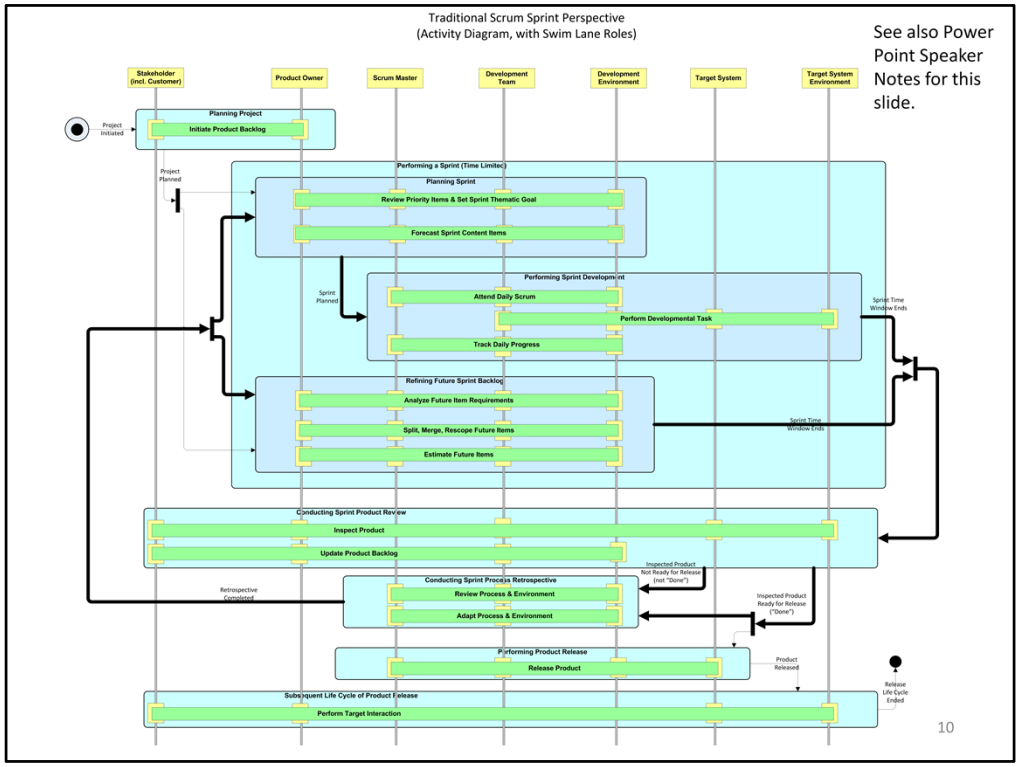


- Traditional agile methods focus on certain key information, summarized by this information model:
    - The Product Backlog is a list of Backlog Items that are candidates for inclusion in deliveries.
    - Backlog items may be Features for stakeholders, or other (internally prioritized) non-Feature items. (In S*Patterns, the stakeholder model is wide enough that there really are no needs that are not for some stakeholder, so the Feature items should cover everything.)
    - Backlog Items are prioritized by the Product Owner, interacting with the Customer (and other stakeholders).
    - Backlog Items are analyzed by the Development Team to produce more descriptive Requirements, along with estimates of related cost or effort.
    - Since Deliverables can (as appropriate) include Design, Test Results, etc., there can be more ISO15288 content in this simple model than first appears.
    - Based on priorities and estimates Backlog Items are chosen to include in a Sprint.
    - During the Sprint, team members record daily progress, including updates to how much effort will apparently be needed to complete the sprint, tracked in the Sprint Burndown.

Traditional Scrum Sprint Perspective
(Activity Diagram, with Swim Lane Roles)

See also Power Point Speaker Notes for this slide.





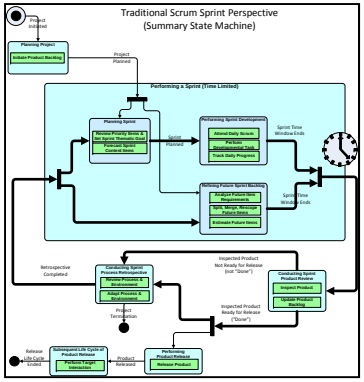
- This diagram represents the same agile process (notice the same process States and Interactions), but emphasizes the key roles associated with agile methods, and who is responsible to perform them—both development roles and actors they encounter:
    - The Customer (more generally, all the stakeholder classes)
    - The Product Owner, who represents all the stakeholders and is responsible for the value delivered to them, beginning with the prioritized Product Backlog (listing of what is to be delivered, in stakeholder Feature terms)
    - The Development Team, intentional very small (~7 or fewer), and responsible for all the technical work necessary to deliver the product, including (as needed) requirements analysis, design, acquisition, construction, test/verification, etc.
    - The Scrum Master, who serves the Development Team to make their work possible and productive.
    - The Development Environment, including laboratory or other technical facilities, test stands, and the general work environment.
    - The Target System to be developed or modified by the delivery process.
    - The Target System Environment, which it will encounter over its subsequent life cycle.

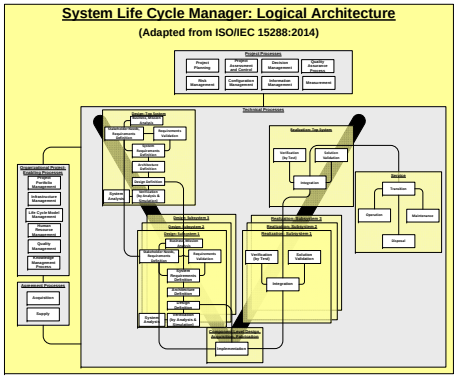
More Than One Representation (Model View) of the Same Underlying Reality

We are dealing with four different representations of the same underlying reality:

1. The Scrum Pattern: Emphasizes time and feedback, focusing on processes for learning and management of risk
2. The ISO15288 Pattern: Emphasizes types of processes, focusing on management of processes
3. The Agile Systems Engineering Life Cycle Pattern : Shows how (1) and (2) above may be seen as one
4. The S*Metamodel: Emphasizes the information flowing through all three of them: (1), (2), and (3)

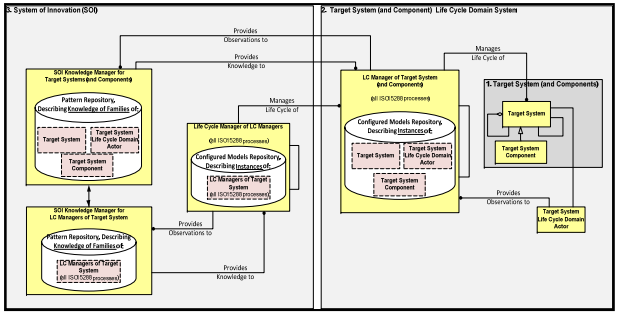
Scrum Pattern

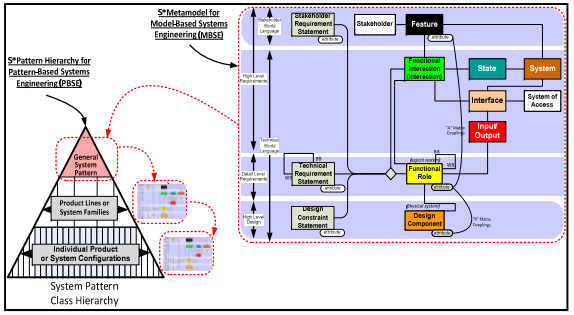




ISO15288 Pattern

ASELC Pattern



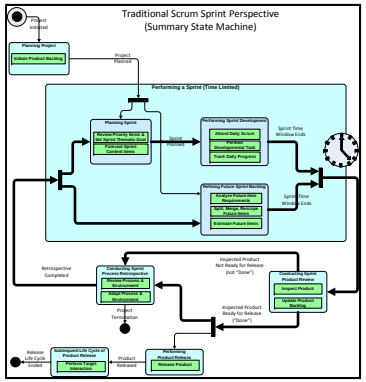


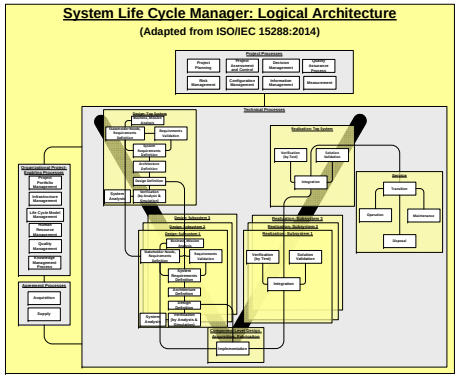
S*Metamodel

More Than One Representation (Model View) of the Same Underlying Reality

- The Scrum Model is actually an abstraction of the more complex-looking multiple Processes of the ISO15288 System Life Cycle reference model:
  - As indicated in the Agile literature, nothing about the Scrum Model is intended to prevent things like Requirements Analysis, Verification (Test), or even aspects of Project Management, . . .
  - But those activities are shared by the small team members who play many individual roles, and the simpler-looking Scrum model "gives us permission" to "do what is needed" in a given situation, in an "agile way".

Scrum Pattern





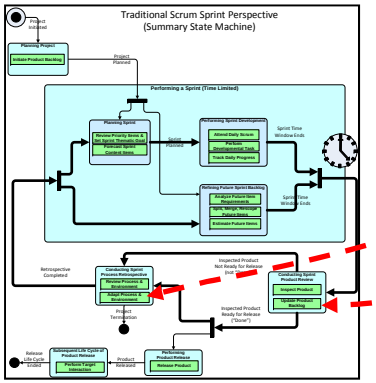
ISO15288 Pattern

More Than One Representation (Model View) of the Same Underlying Reality

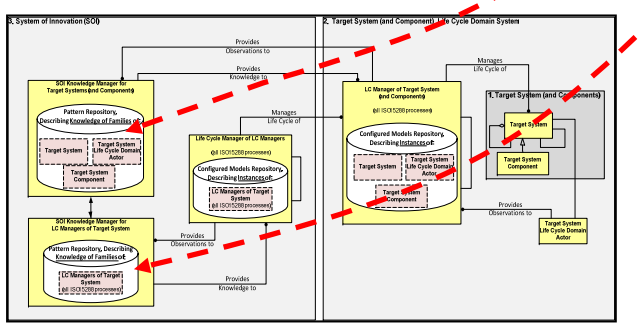
- The Scrum Model also abstracts complex learning behavior, into simple-looking form—but it is still strongly expected to occur as part of the Agile Process, and is more explicitly represented in the ASELC Pattern, as capture of Pattern information.

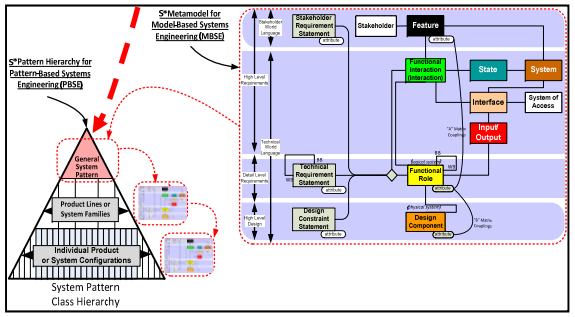
Learning

Scrum Pattern



ASELC Pattern

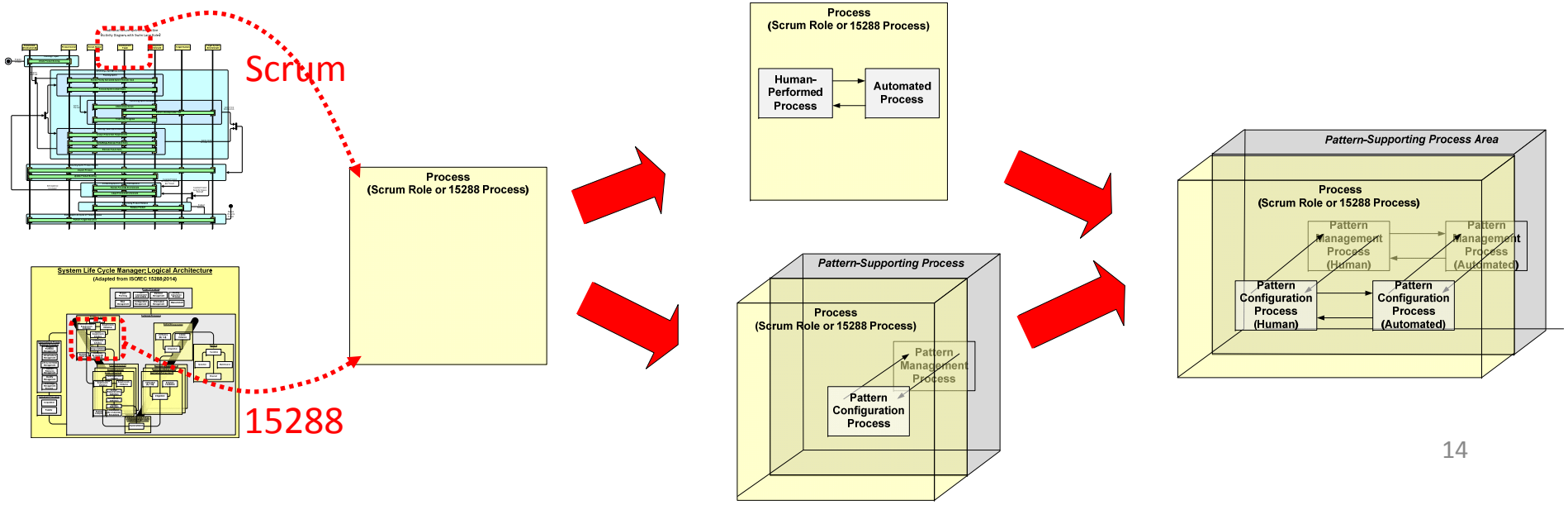




S*Metamodel

More Than One Representation (Model View) of the Same Underlying Reality

- Although the Scrum Model purposely simplifies those complexities, for ease of understanding “agility”, a complete analysis of the Agile SE Process requires that the process analyst understand their formal representation in the context of the Agile SE Process Model.
- This includes how each of the Scrum Model and 15288 Model “roles” can be decomposed two different ways:
  - (Human Agent) interacting with (Automated Information System Agent)
  - (Management of Current System Configuration) interacting with (Management of Pattern)
- Resulting in four interacting internal roles:



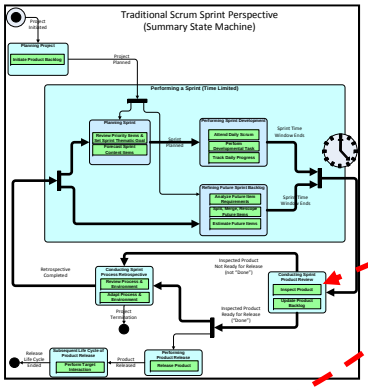
More Than One Representation (Model View) of the Same Underlying Reality

- Notice that the division of the System 3 roles in the ASELCM Pattern corresponds to the Scrum division of (review and learning about target system) versus (review and learning about development process):

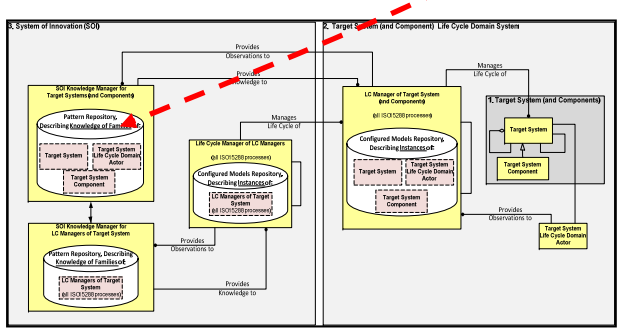
Pattern: Learnings about Target System (Product & Its Environment)

Pattern: Learnings about Development / Fielding System & Its Environment

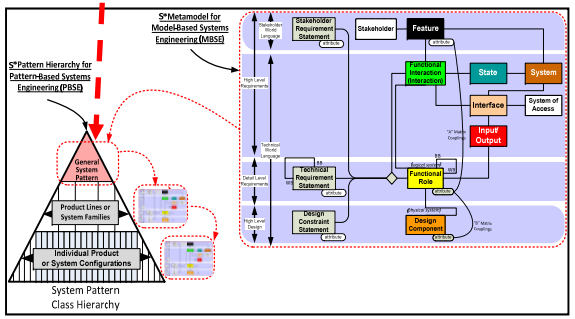
Scrum Pattern



ASELC Pattern



S*Metamodel

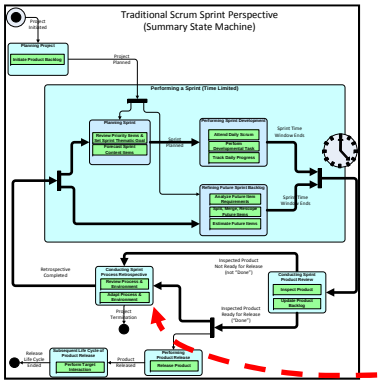


More Than One Representation (Model View) of the Same Underlying Reality

- Notice that the division of the System 3 roles in the ASELCM Pattern corresponds to the Scrum division of (review and learning about target system) versus (review and learning about development process):

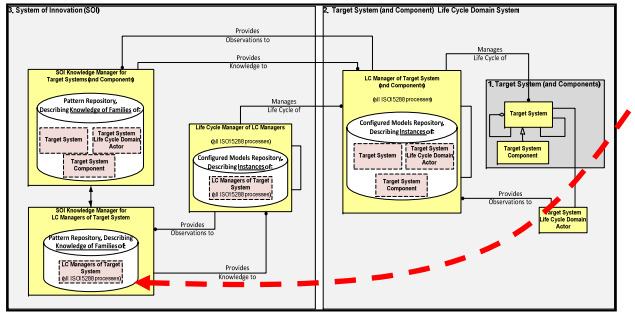
Pattern: Learnings about Target System (Product & Its Environment)

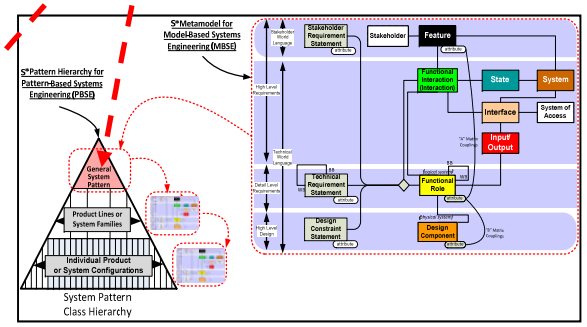
Scrum Pattern



Pattern: Learnings about Development / Fielding System & Its Environment

ASELC Pattern





S*Metamodel