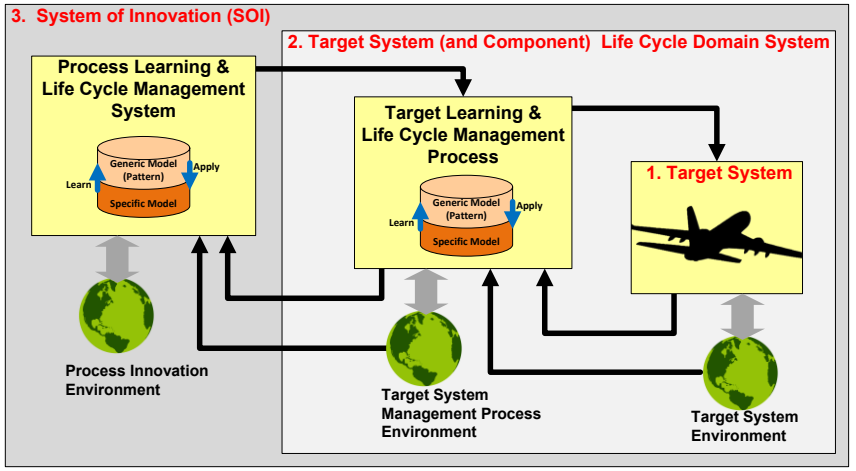
ICTT System Sciences®
Understand your systems.

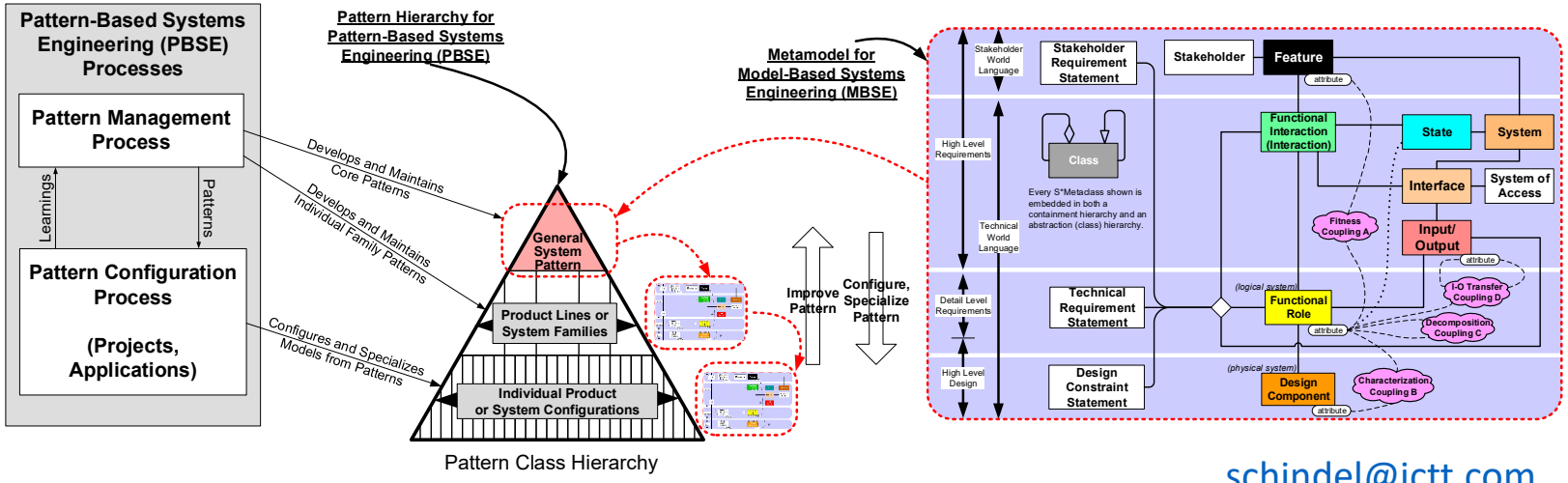
Phase I Session 4

<<Company/Org. Logo>>

Introduction to S*Models, S*Metamodel, S*Patterns, Mappings to Languages & Tools, S*Configuration Wizard, ASELCM S*Pattern



Level 0 View, INCOSE ASELCM Pattern

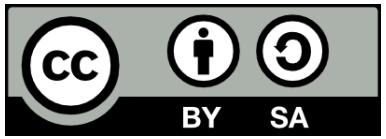


Pattern Class Hierarchy

schindel@ictt.com
lewis@ictt.com

V1.2.11

Copyright 2025, System Sciences, LLC



Licensed under a Creative Commons
Attribution Share Alike-License CC BY SA International 4.0

License Link: https://creativecommons.org/licenses/by-sa/4.0/legalcode

Uses are permitted under this license without further permission from the copyright owner, provided each use (1) is clearly marked to attribute the underlying work to "S*Patterns Community", (2) provides a link to the CC BY SA license, (3) indicates if changes were made, (4) does not suggest the licensor endorses the user or use, (5) does not apply legal terms or technological measures that legally restrict others from doing anything the license permits, and (6) if you remix, transform, or build upon the material, you must distribute your contributions under the same license as the original.

Permissions beyond the scope of this license are administered through contacting:

Corporate Officer
ICTT System Sciences
378 South Airport Street
Terre Haute, IN 47803
812-232-2208

Systematica is a registered trademark of System Sciences, LLC.

Contents

- Recap of Session 3:
  - Session 3 Take-aways, discussion, questions
  - Session 3 Homework
- Phase I Session 4 Class Content
- References

Recap of Session 3:
Take-aways impacting actual practices

What consistent practices seem possible to this team, across different systems, clients, and projects?

1. Representing **Failure Modes** ("FM" of FMECA) as "abnormal states" in which a system component's behavior departs from Requirements normally satisfied?
2. Representing that abnormal behavior as **Counter Requirements**?
3. Representing the consequence/effects ("E" of FMECA) of that bad behavior as feature-inverting **Feature Impacts**, having Severity attributes?
4. Representing **Risk** as ~ [Severity of Impact of Effect of Event] x [Likelihood of Event] x [Detectability Score] x [Mitigation Score]?
5. Generating **Fault Tree Analysis (FTA)** versus **Failure Mode Effects and Criticality Analysis (FMECA)** views from same risk model data?



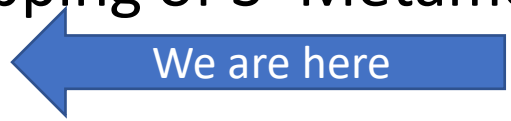
- Can a creative, talented team *align* its work?
- What should be *standard* about this, in your customer work?
- What should be *configurable* about this, in your customer work?



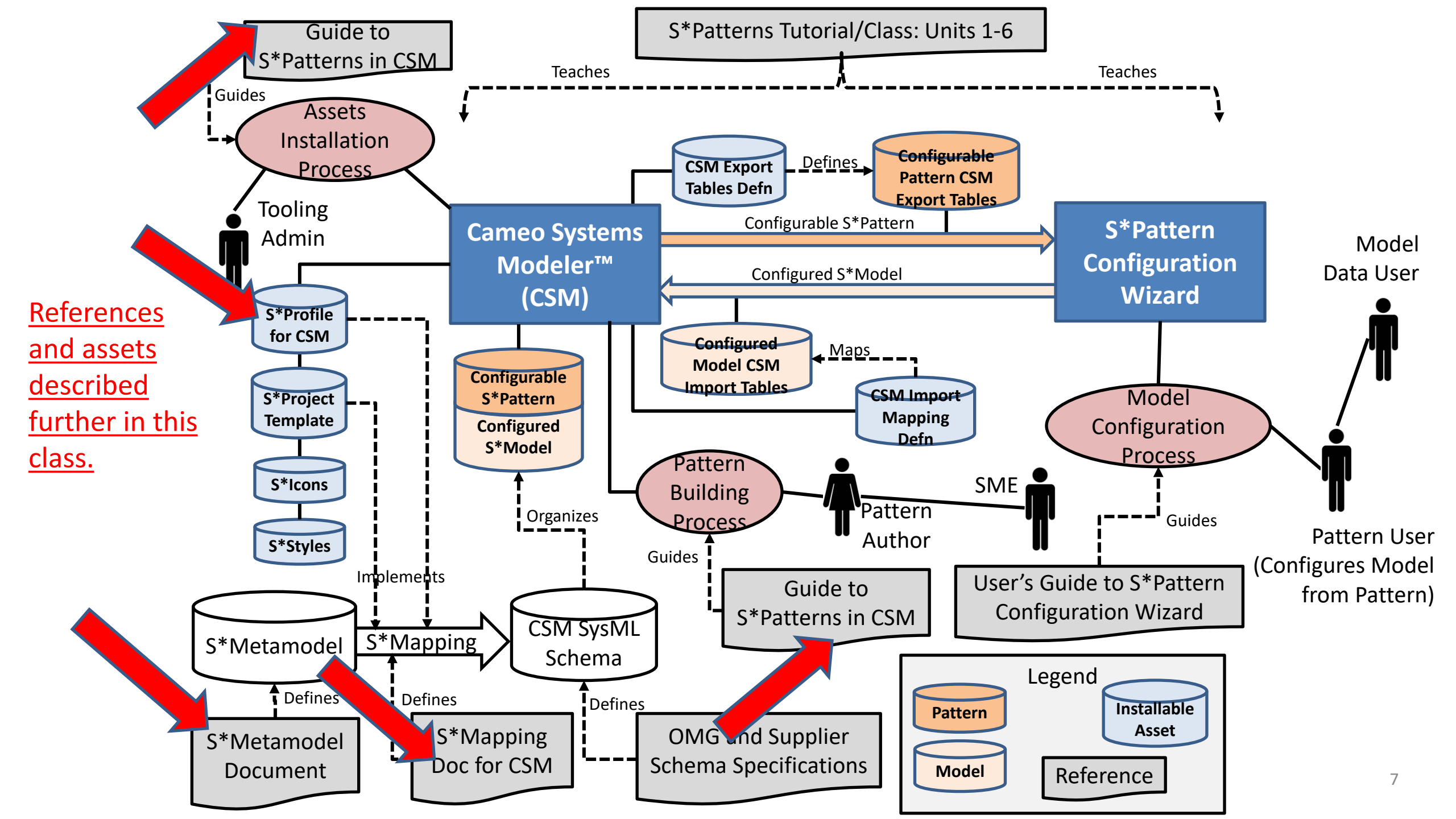
Session 3 Homework Discussion

- Assignment was:
- For the example pattern you have been using for HWs 1 & 2, add metarelationships to the FMEA elements added during the in-class exercise:
- Impacts Feature (Pattern Failure Impacts Table)
- Causes Impact (Pattern Failure Impacts-Counter Requirements Table)
- Replaces (Pattern Counter Requirements Table)
- Causes Behavior (Pattern Failure Modes-Counter Requirements Table)
- Abnormal State Of (Pattern Failure Modes Table)
- Provides Failure Context (Pattern Failure Mode Context Table)
- Provides Failure Mode Context (Pattern Failure Mode Context Table)
- Bonus: Show some RPN scores, based on the mode probability score, severity score, and detectability score.

Phase I Class Sessions: Microstructure of Models and Patterns (6 Sessions, 3 hours each)

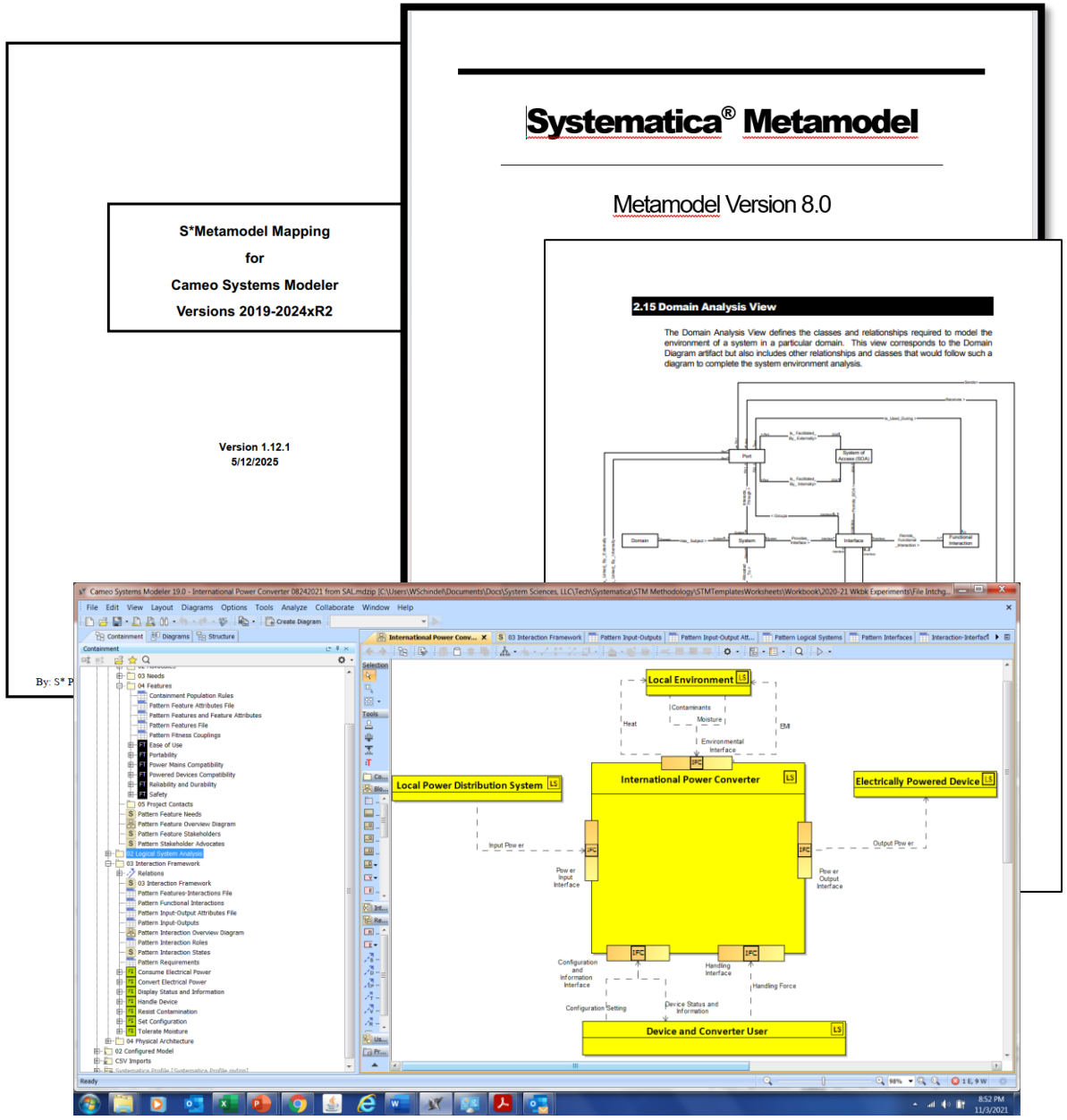
- S*Models, S*Metamodel:
  - Sessions 1-3
  - In Example Model Context
- Formal mapping of S*Metamodel into Tooling
  - Session 4 

- S*Patterns, Configuration: In Example Model Context
  - Session 5
- Introduction to 3rd Generation Pattern Configurator Technology
  - Session 6
- Homework Presentation and Feedback
  - Distributed over above sessions


Guide to S*Patterns in CSM
S*Patterns Tutorial/Class: Units 1-6
Teaches
Teaches
Guides
Assets Installation Process
Tooling Admin
CSM Export Tables Defn
Defines
Configurable Pattern CSM Export Tables
Cameo Systems Modeler™ (CSM)
Configurable S*Pattern
Configured S*Model
S*Pattern Configuration Wizard
Model Data User
References and assets described further in this class.
S*Profile for CSM
S*Project Template
S*Icons
S*Styles
Configurable S*Pattern
Configured S*Model
Configured Model CSM Import Tables
Maps
CSM Import Mapping Defn
Model Configuration Process
Pattern Building Process
Pattern Author
SME
Organizes
Guides
Guides
Implements
Pattern User (Configures Model from Pattern)
S*Metamodel
S*Mapping
CSM SysML Schema
Guide to S*Patterns in CSM
User's Guide to S*Pattern Configuration Wizard
Defines
Defines
Defines
S*Metamodel Document
S*Mapping Doc for CSM
OMG and Supplier Schema Specifications
Legend
Pattern
Installable Asset
Model
Reference

Formal mapping of S*Metamodel into Tooling

- Semantics and interoperability
- Semantics of models
- Purpose of S*Metamodel and its mappings
- Mapping into standard languages, third party tools, enterprise infosystems
- Examining the form of a formal mapping
- A tool specific profile; installing and navigating
- Applications to patterns, frameworks, ontologies, schema
- Other tools and semantic frameworks; semantic technologies; OWL; ST4SE project
- Homework assignment 4
- Discussion, Q&A.



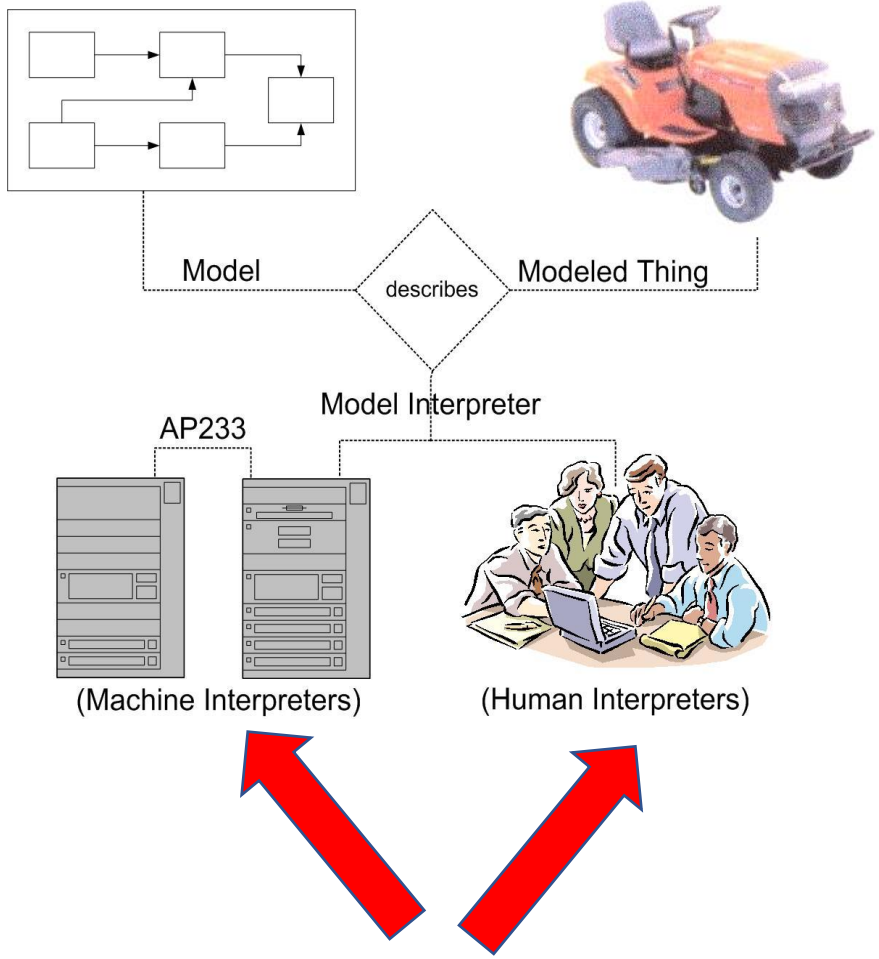
Semantics and interoperability



- In the general popular culture, the term “semantics” has a “bad name”:
  - “It’s just semantics” is a common pejorative reference, implying some sort of irrelevant, hair-splitting distinction that is not worthy of attention.
  - But the joke is on the teller . . .
  - “Semantics” is about *meaning*.
  - If you are willing to disregard the meaning of what you say and hear, and just “go ahead without it”, we wish you good luck.
- Shared Meaning: If my conversation partner does not know what I mean, assigning a different meaning to my words than I do, our conversation is at risk:
  - But let’s not fool ourselves . . . . we all know that this is the case to some limited degree in virtually all our conversations.
- What does this have to do with MBSE models?

Semantics of models

- Two subject matter experts will frequently have different interpretations of a carefully written prose specification.
- But such disagreement is very rare if they are reading an electrical schematic or mechanical drawing. Why? — Because of their formal semantics and enforcement.
- Formal models allow us to “tighten up” the meaning (semantics) of our representations:
  - Compared to everyday conversation in natural language.
  - Even compared to “technical” but less formal slide presentations.
  - So we communicate with others more successfully.
- Models also promise:
  - Remembering the success of models in the STEM revolution,…
  - Not only might we better understand each other, but <u>also</u> we can more accurately describe the natural and human-made world.
  - There are many kinds of models, much older than MBSE, that have long worked extremely well for their intended purposes.



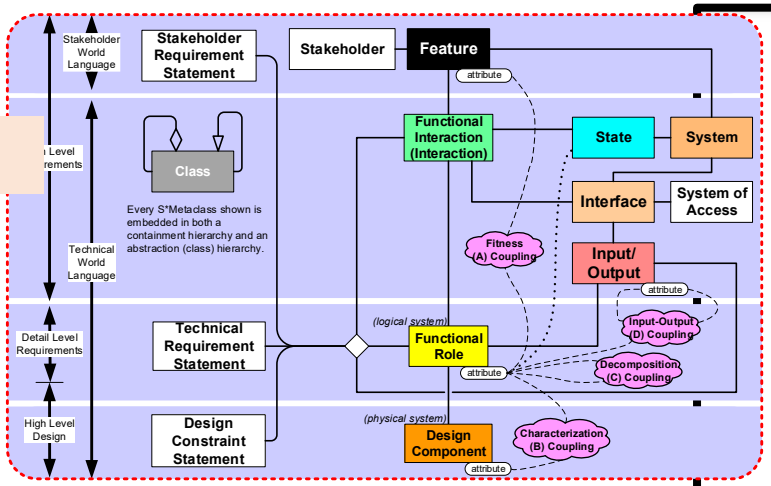
<u>Model interpreters</u>: Where model semantics are applied

S* (Systematica) Metamodel

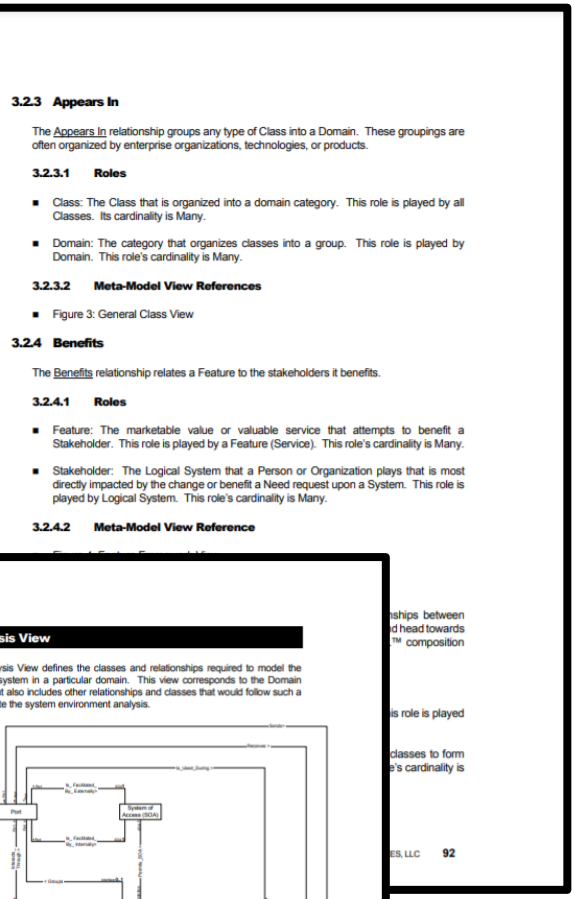
Remember the last 20 years' goal for the S*Metamodel:

- Describe the smallest set of model elements necessary for the purposes of engineering and science, over the life cycle of systems.
- Independent of tooling or other languages--*until mapped into them!*
  - Part of today's session.

Informal



S*Metamodel informal summary pedagogical diagram
(formal S*Metamodel includes additional details.)



3.2.3 Appears In

The Appears In relationship groups any type of Class into a Domain. These groupings are often organized by enterprise organizations, technologies, or products.

3.2.3.1 Roles

- Class: The Class that is organized into a domain category. This role is played by all Classes. Its cardinality is Many.
- Domain: The category that organizes classes into a group. This role is played by Domain. This role's cardinality is Many.

3.2.3.2 Meta-Model View References

- Figure 3: General Class View

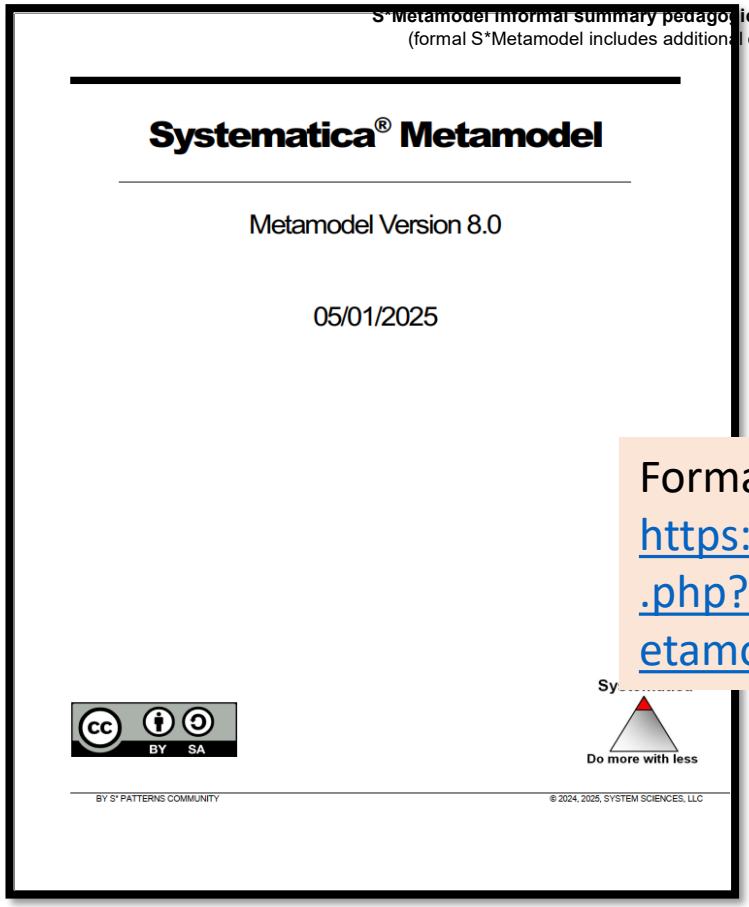
3.2.4 Benefits

The Benefits relationship relates a Feature to the stakeholders it benefits.

3.2.4.1 Roles

- Feature: The marketable value or valuable service that attempts to benefit a Stakeholder. This role is played by a Feature (Service). This role's cardinality is Many.
- Stakeholder: The Logical System that a Person or Organization plays that is most directly impacted by the change or benefit a Need request upon a System. This role is played by Logical System. This role's cardinality is Many.

3.2.4.2 Meta-Model View Reference



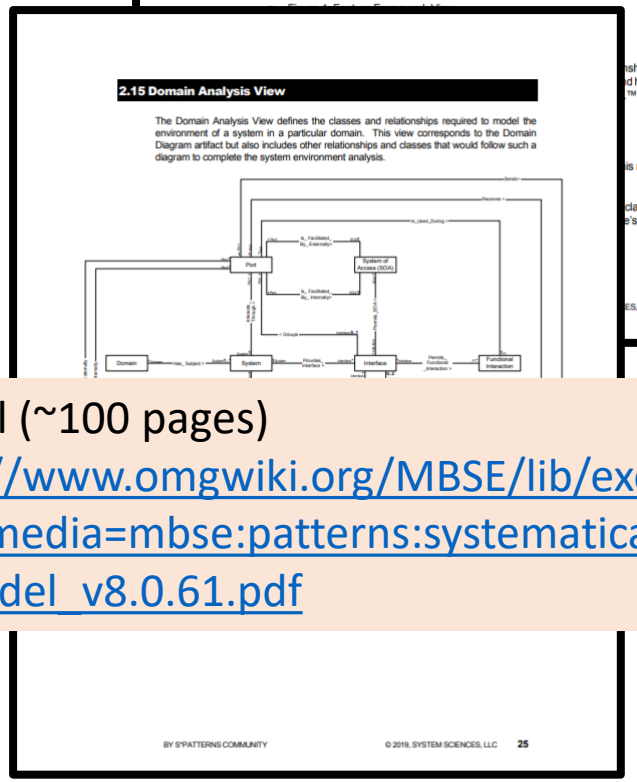
Systematica® Metamodel

Metamodel Version 8.0

05/01/2025

Do more with less

BY S* PATTERNS COMMUNITY © 2024, 2025, SYSTEM SCIENCES, LLC



2.15 Domain Analysis View

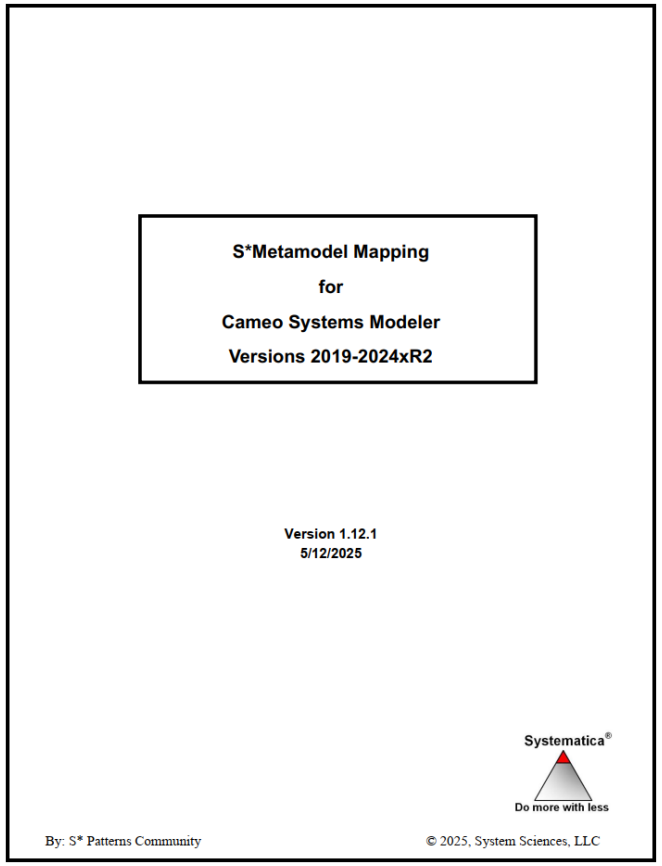
The Domain Analysis View defines the classes and relationships required to model the environment of a system in a particular domain. This view corresponds to the Domain Diagram artifact but also includes other relationships and classes that would follow such a diagram to complete the system environment analysis.

BY S*PATTERNS COMMUNITY © 2019, SYSTEM SCIENCES, LLC 25

Formal (~100 pages)
https://www.omgwiki.org/MBSE/lib/exe/fetch.php?media=mbse:patterns:systematica_8_metamodel_v8.0.61.pdf

Mapping of S*Metamodel to Cameo Systems Modeler (OMG SysML tooling)

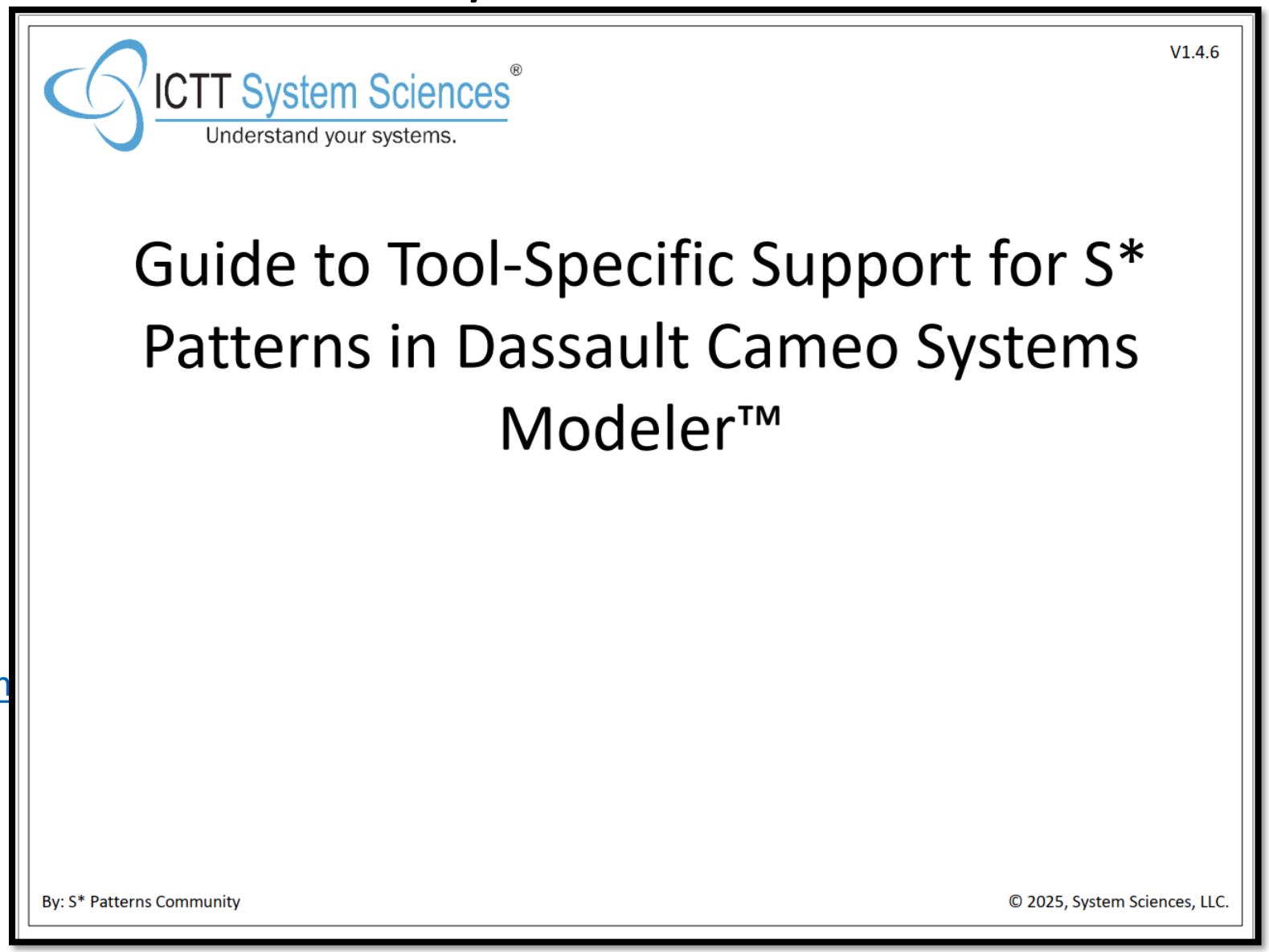
- Illustrates a typical mapping to a third-party COTS modeling tool schema—in this case, a SysML modeling tool.
- Establishes basic semantic interoperability for the tool with other S*Metamodel mapped tools, information systems, *or humans.*
- Later in today's class session.



https://www.omgwiki.org/MBSE/lib/exe/fetch.php?media=mbse:patterns:systematica_mapping_for_magicdraw_csm_v1.12.1.pdf

S* Patterns Guide for Cameo Systems Modeler

- Describes how to implement and use the mapped S* Metamodel in Cameo Systems Modeler™
- Later in today's class session.



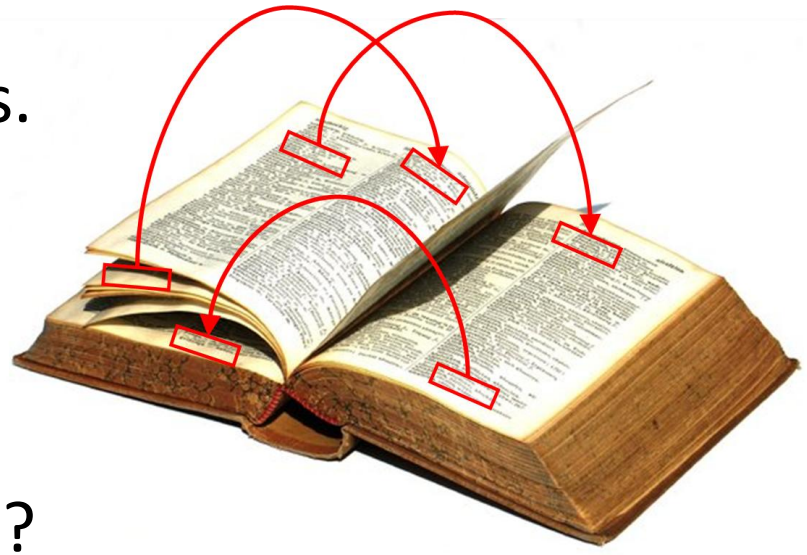
https://www.omgwiki.org/MBSE/lib/exe/fetch.php?media=mbse:patterns:guide_to_systematica_patterns_in_cameo_systems_modeler_v1.4.6.pdf

Other historical S*Metamodel mappings

- Telelogic / IBM DOORS
- OMG SysML
- IBM Rhapsody SysML
- Sparx Systems Enterprise Architect (EA)
- No Magic/Dassault Systems Cameo Systems Modeler (CSM)
- Vitech GENESYS
- IBM Requisite Pro
- Dassault Enovia PLM
- Siemens Team Center Systems Engineering

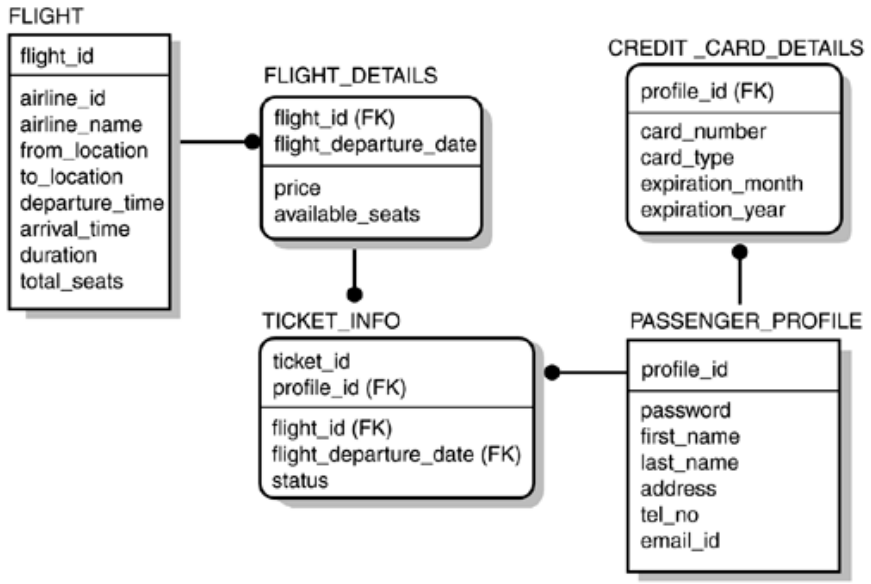
Warm Up: The Unabridged Dictionary Thought Experiment

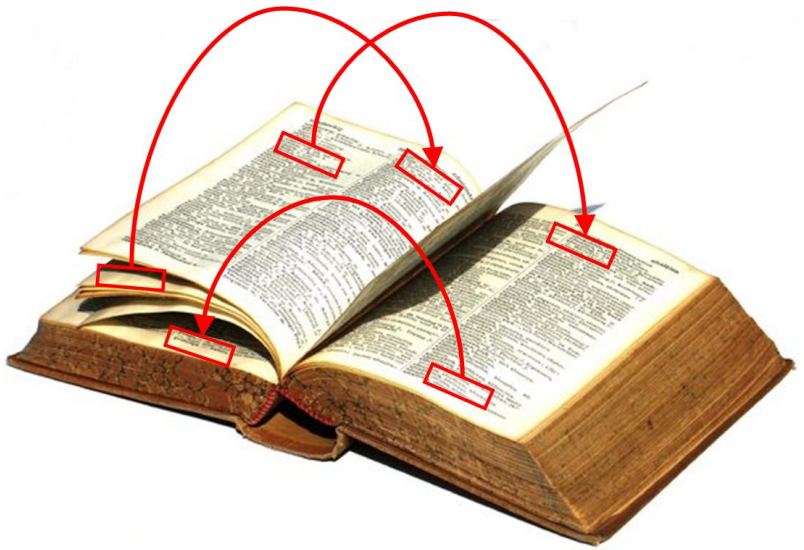
- Here is an Unabridged Dictionary—all of its “definitions” are limited in their use of words to those which are themselves defined in the same book.
- Imagine that we insert connecting threads (these days, we’d call them “hyperlinks”) that connect every word in every definition to its own definition.
- Allow no exceptions—every word.
- Suppose it also contains a section on grammar rules.
- The grammar rule words are likewise connected.
- Now imagine that all the type fades away.
- Only the connections to empty nodes remain.
- Do you believe that we still have all the information?
- The “labels” are lost, and we could replace them with other words.
- Would the resulting publication (and language) still “mean” the same thing?



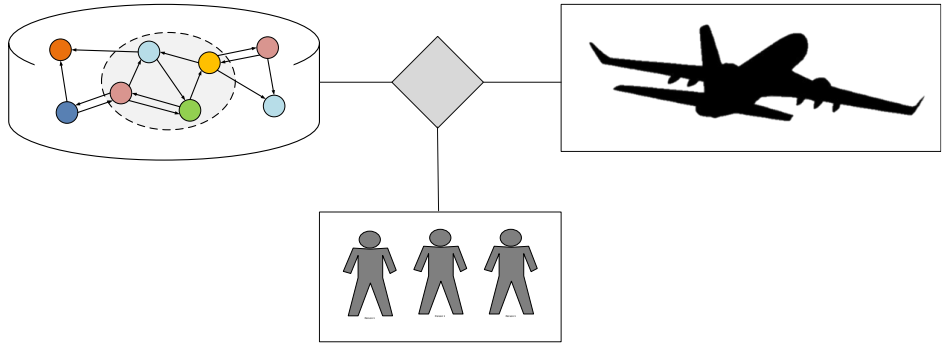
Warm Up: The Unabridged Dictionary Thought Experiment

- This thought experiment suggests the nature of “relational” models:
  - Models in which modeled relationships between “things” tells us the main things (in some cases, everything) we want to say or know about them. (Their “prose definitions” are formally unimportant.)
- This brings up the interesting question about including word (prose) definitions of the things in our models, for the “practical” reason of others understanding them:
  - Are the word definitions claiming things about the modeled elements that are not reflected in the rest of the model’s connections and structure?
  - Ontologists (who model ontologies—frameworks of meaning for other models) sometimes argue about whether word definitions of the classes in their ontologies are helpful, misleading, or conflict with the relational model.

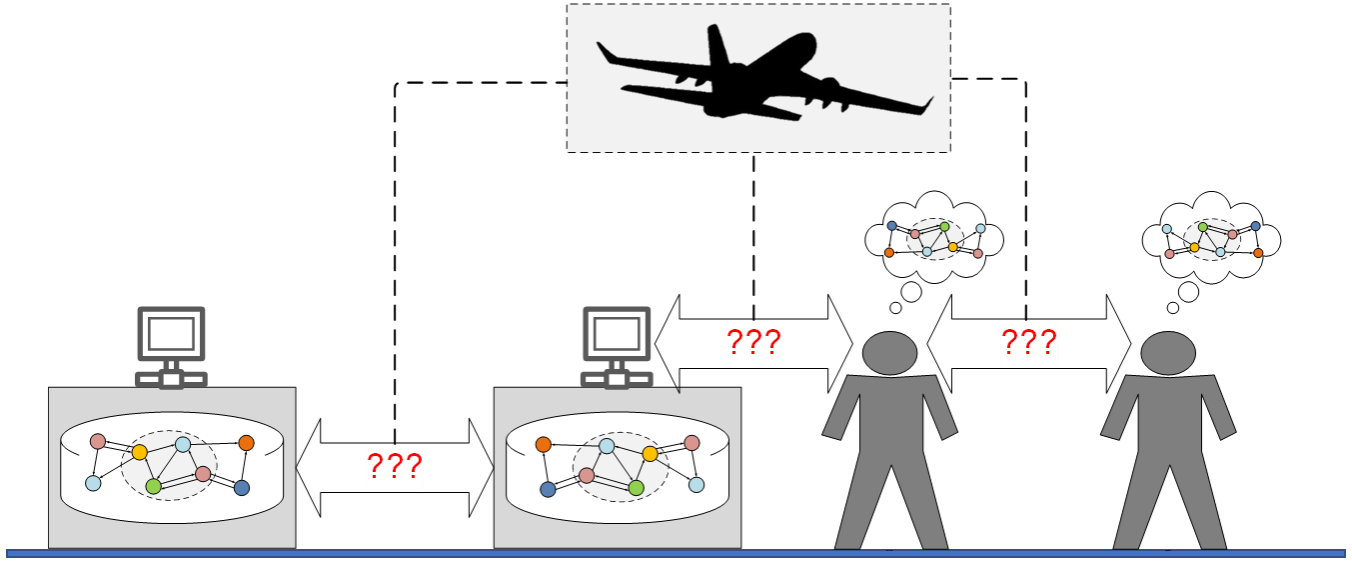




Semantic inter-operability



- The setting for this discussion:
  - Generation and use of computer-based virtual models continues to grow across industry.
  - Likewise, humans also internalize their own mental “models” of systems of interest.
  - Needs for “Interoperability” across all these models (including people) can be a challenge.
  - This briefing summarizes the underlying nature of that challenge and its solutions.
  - Some of these aspects are well-known, but certain important aspects sometimes missed.



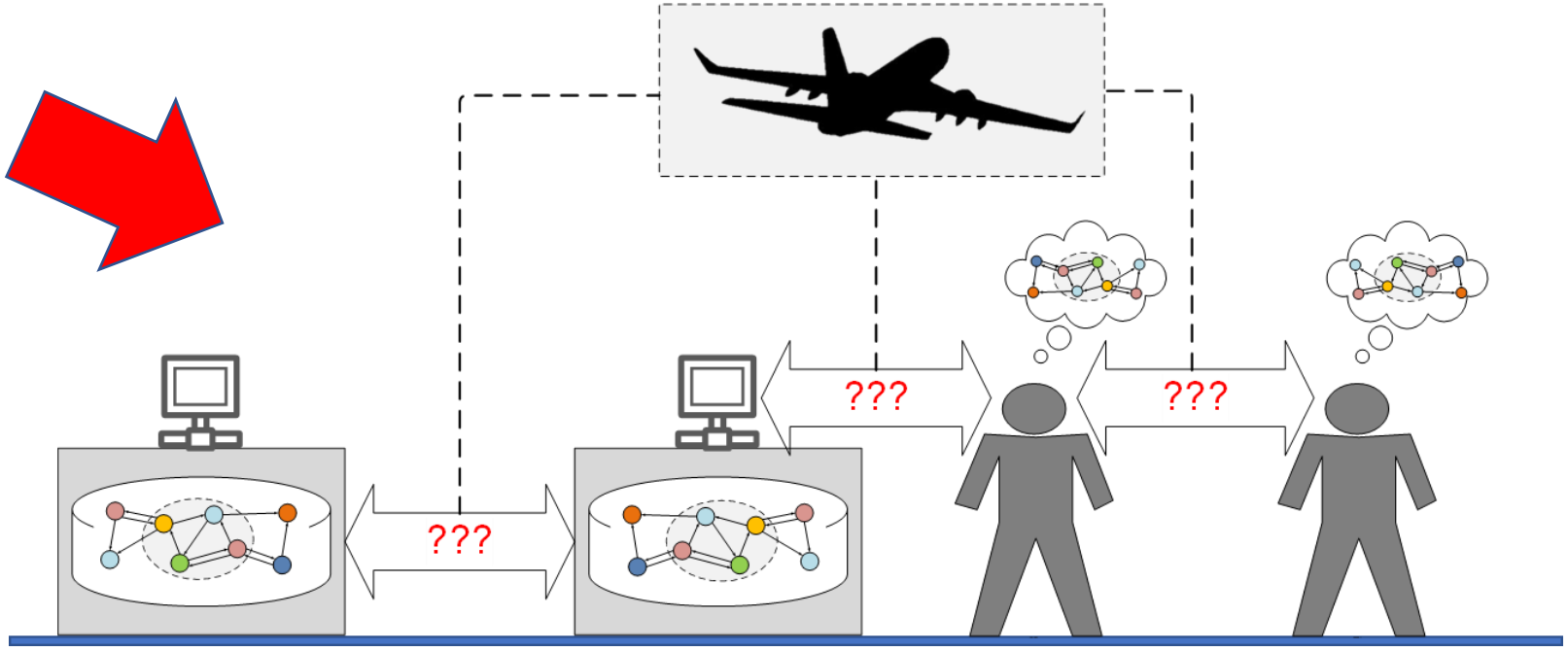
“Playing well together”

- Impact on ability to “work together effectively”, across different individuals, teams, organizations, specialties, capabilities, businesses, etc.
- Includes crossing boundaries across not just between automated tools, but also between different humans individuals/teams, and between tools and people.

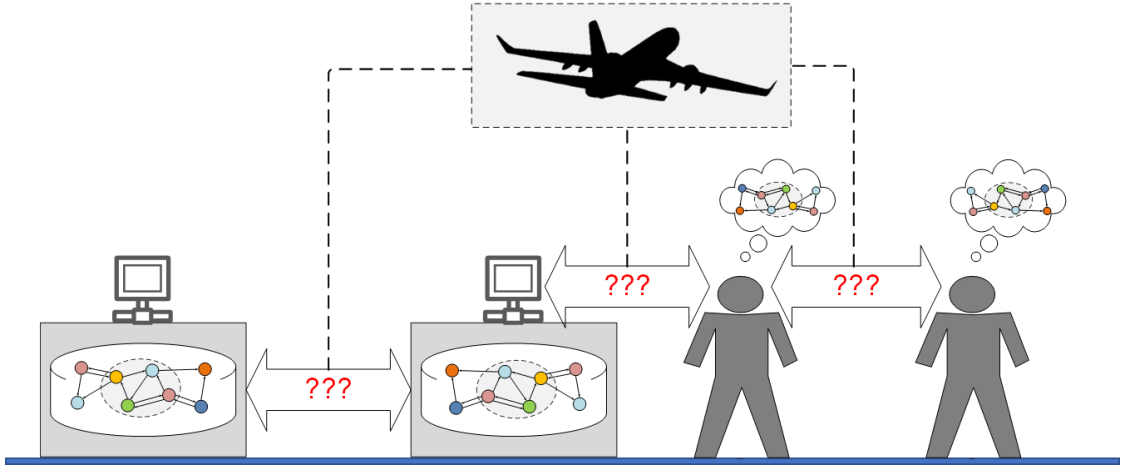
We suggest that the real implications of this simplistic diagram are not as well understood as needed.

What does it really mean?

Why does it matter?



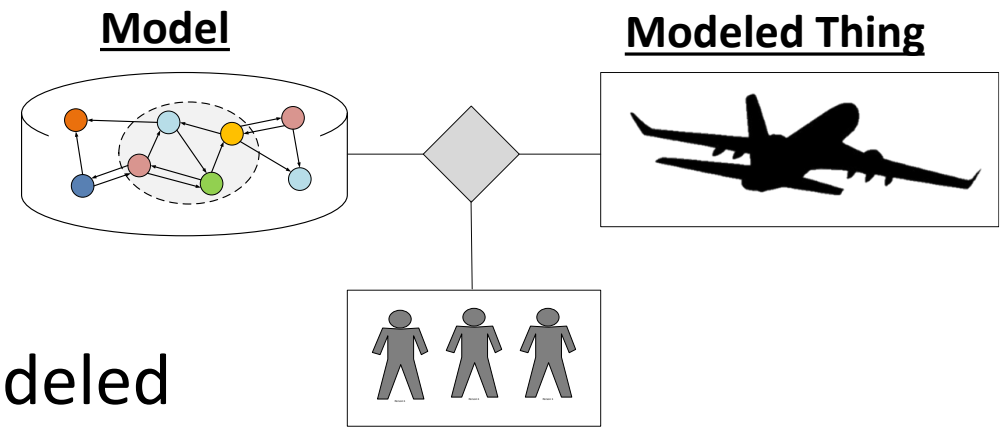
Picking the right problem



- We often hear complaints that various model-based tools and database platforms are “incompatible” with each other, in the sense that they cannot do some kind of “data exchange”.
- This is not a very helpful way to understand the challenge, as it appears to suggest some kind of IT problem that should be solved by automation vendors.
- Simple example: I have a question that requires use of information from several tool chain components. How hard is it for me to query for an answer? Do I have to do the integration in my head, or do the tools help me?
- Bigger picture: What is the overall need? What is the problem of interoperability?

What is meant by “semantics” of models?

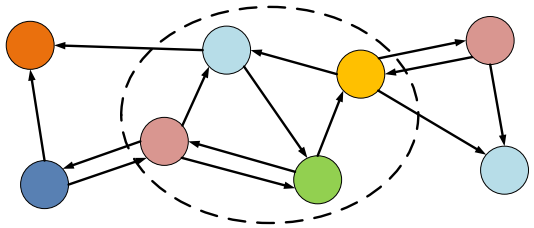
- This diagram is the right place to start:



- The Model is said to describe aspects of the Modeled Thing *to the Model Interpreters* (Model Users).
- The Model is said to be “about” the Modeled Thing.
- We need to understand model “about-ness”.
- Model semantics is what encodes that “about-ness”.
- This idea is universal to all types of virtual models.
- What does model semantics look like? . . . .

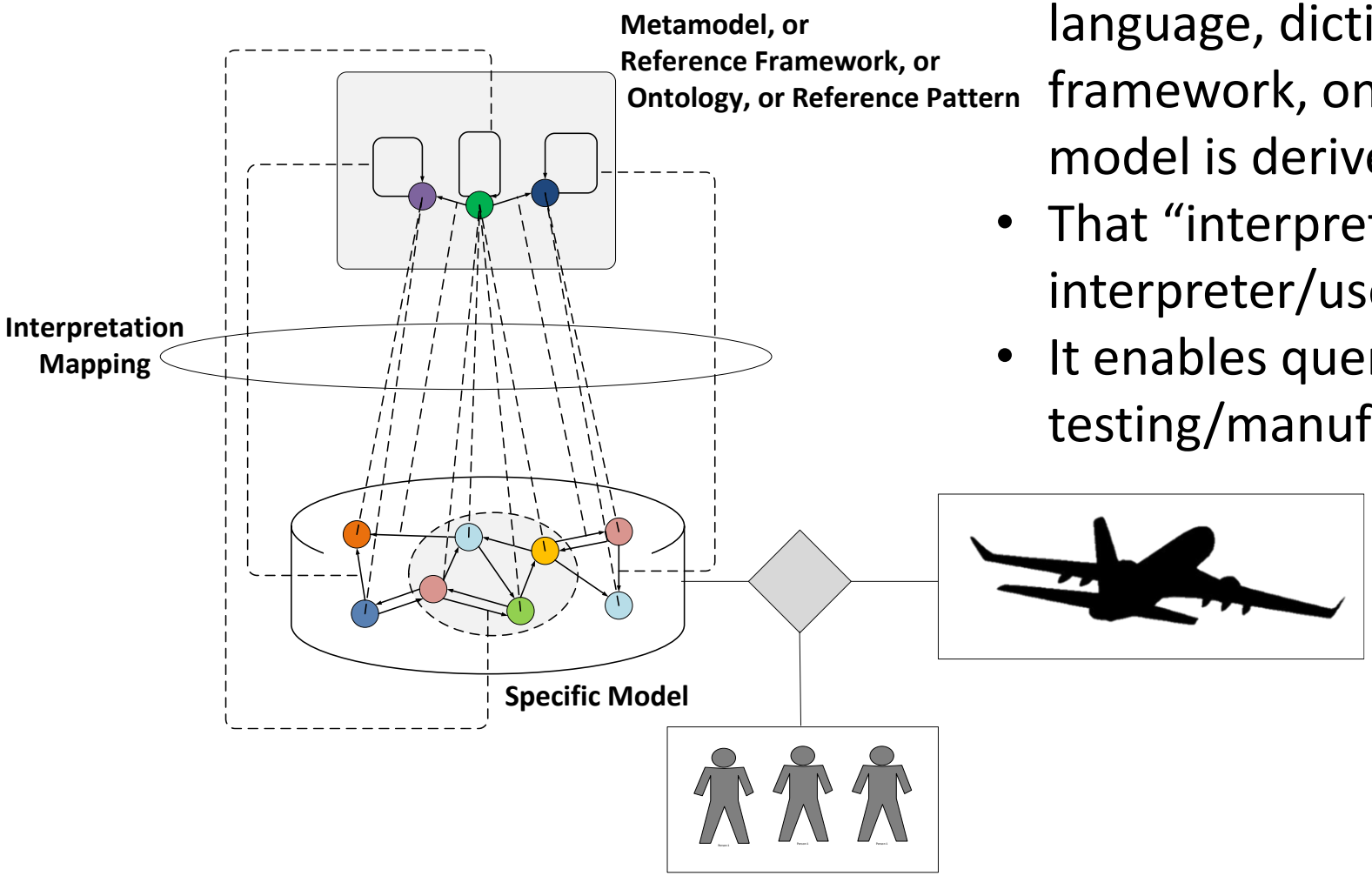
The web of meaning

- Different model types, languages, and tools differ in detail, but . . .
- The core “semantics” of all those models are effectively encoded as information:
  - A web of relationship links (connections)—whether symbolic or graphical.
  - The nodes that are connected.
  - The nodes and relationships can have names (including values in some cases).
- The names seem to attract more attention, but are not really the main challenge of model semantics.
- The essence of the semantics are expressed by:
  - The shape of the web of links in the model, and . . .
  - The trace of those nodes and links of a model to a more general “reference” web that expresses the language, dictionary, template, architectural framework, ontology, or pattern from which the model is derived or expressed.



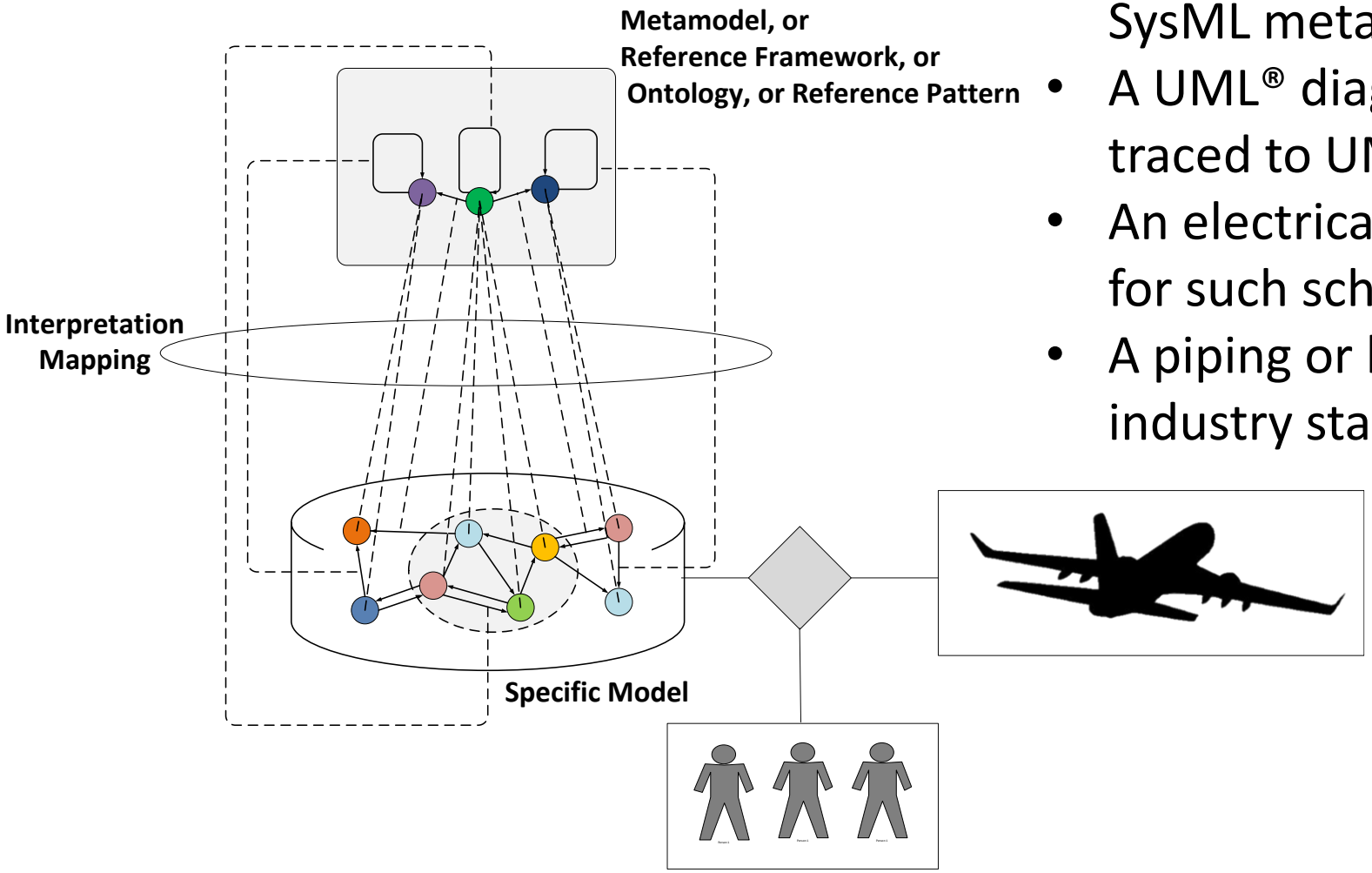
The essence of the model's semantics are expressed by:

- The web of named nodes & links in the model.
- The trace of those nodes and links of a model to a more general "reference" web that expresses the language, dictionary, template, architectural framework, ontology, or pattern from which the model is derived or expressed.
- That "interpretation" trace tells the model interpreter/user how to read/interpret the model.
- It enables queries, traces, views, reasoning, logic, testing/manufacturing parts, other responses.



Examples:

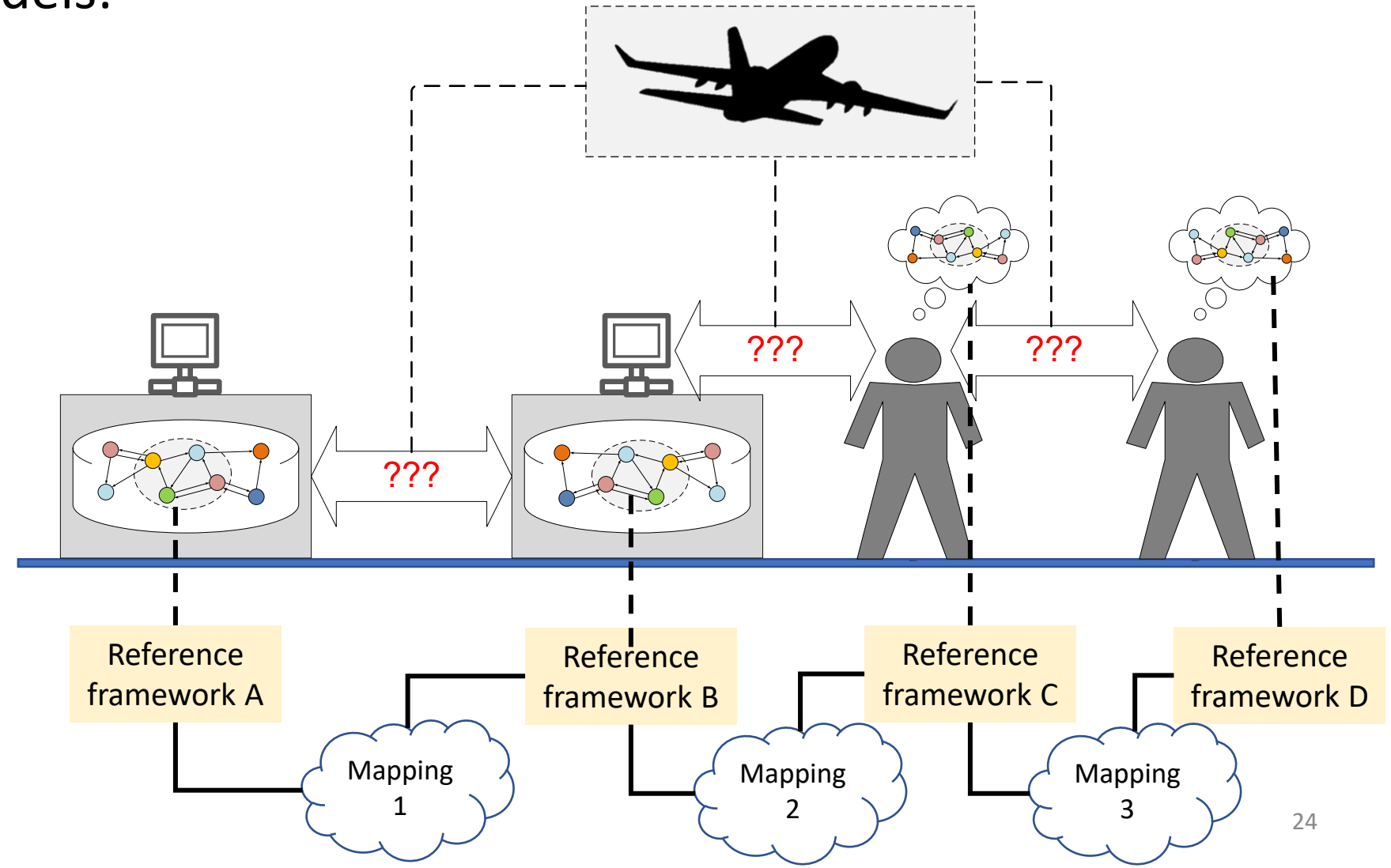
- A Simulink™ simulation model diagram, traced to library of Simulink™ computational blocks;
- A SysML™ model diagram, traced to OMG SysML metamodel or OMG UAF schema;
- A UML® diagram about a computer program, traced to UML metamodel library;
- An electrical schematic, traced to IEC standards for such schematics;
- A piping or hydraulic diagram, traced to industry standards for such diagrams.

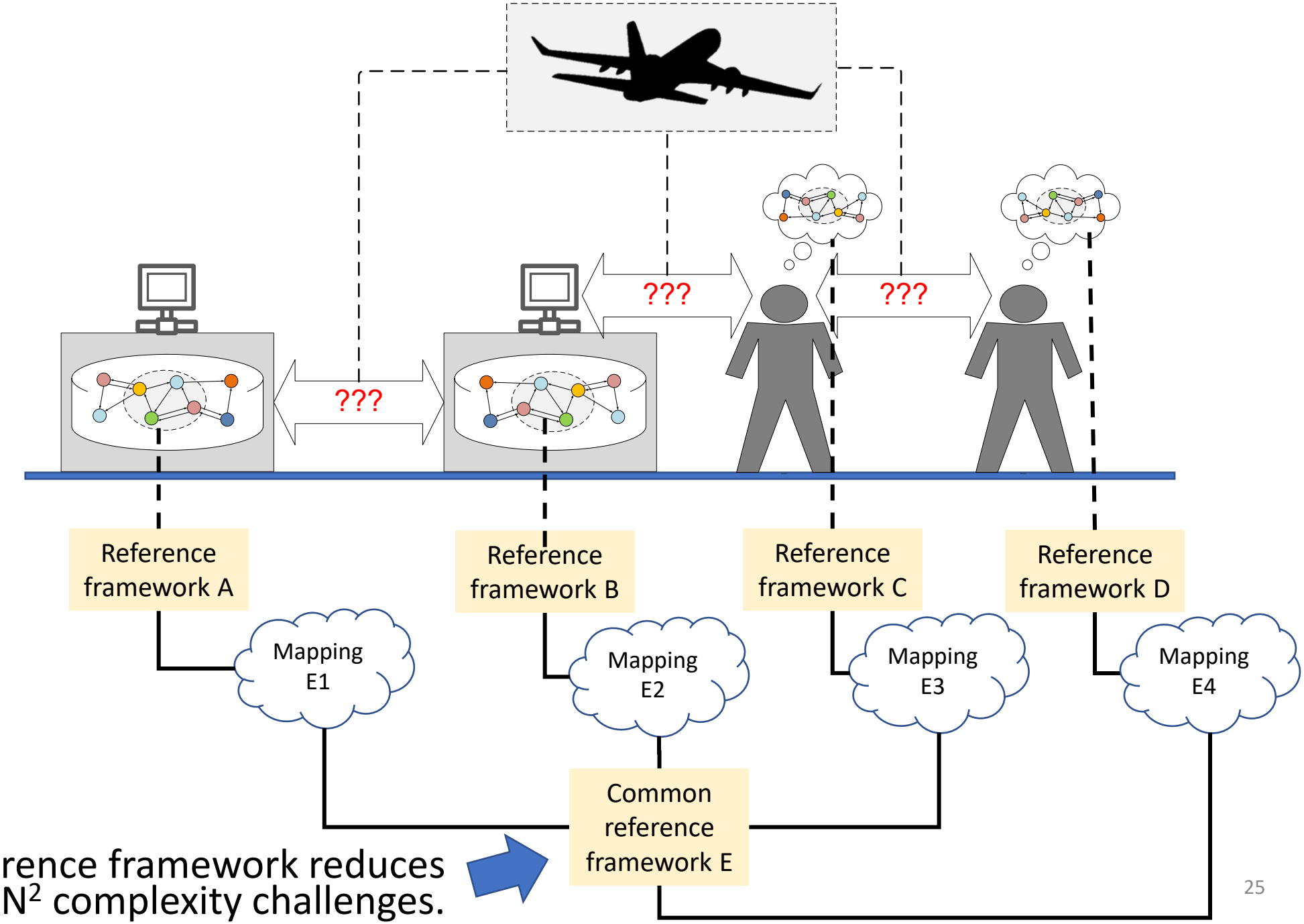


Semantic interoperability

- *Mappings between the reference frameworks*, if they are feasible, are key to interoperability of models:

This diagram understates the typical problem by showing only some "pairwise" mappings. The worst case for that approach is quadratic (N^2) in the number of platforms and people, which is probably overstated.





A “common” reference framework reduces potential N^2 complexity challenges.

Related efforts, constructs, resources: Sampling examples

- Modeling tool language metamodels:
  - OMG SysML Metamodel
  - OPM Metamodel
  - Capella Metamodel
  - Team Center PLM Data Model
- Standards based exchange:
  - ISO 10303 AP233
  - OMG XMI
  - NAFEMS FMI
- Reference metamodels:
  - S*Metamodel
- Reference ontologies
  - Basic Formal Ontology (BFO)
  - Gene Ontology
  - NASA IMCE Ontology
- Hub interoperation & transport tooling:
  - Phoenix Integration ModelCenter
  - Intercax Syndeia
  - HPC Science Gateways (I.U., Apache Airavata)
- Architectural frameworks and related standards:
  - OMG Unified Application Framework (UAF)
  - ISO 42010
- Reference Patterns, Domain Specific Languages (DSLs):
  - ASELCM Pattern
  - Model Characterization Pattern (Wrapper)
  - Manufacturing Pattern
  - Embedded Intelligence (EI) Pattern
  - Product Line Engineering (PLE) Models
- Schemas:
  - Navy Federated Schema

A Tension: Model incompatibility vs. Model insufficiency

1. **Reality Laboratories**: Physical science (and engineering) for ~300 years has dramatically lifted our lives by pressing physical science models to agree with Nature.
2. **Consensus 'Committees'**: Commerce and information technologists for ~50 years progressed by pressing its information models to agree with each other.

- The above (both needed) are not necessarily in conflict, but sometimes do conflict:
  - See Galileo versus The Inquisition (models of the earth-sun-planetary system).
  - "Conventional Wisdom" is sometimes 'wrong', even if widely agreed upon.
- More recent emphasis on formalized consensus standards for systems modeling languages has in some cases disconnected systems engineering models from (1) above:
  - It is very possible (and currently the case) for standards-based tools to be consistent with each other (interoperable) but inconsistent with the Nature (not so useful) they are expected to represent.
  - Example: Interactions (contrast some SE models to *computational* model emphasis on fidelity in representing phenomena of Nature).
  - These are currently resolved by mappings that formalize consistent representation of missing elements in the same existing third party COTS tools.
  - Perhaps a bigger issue than the "standards" content is the variation in people's interpretation, understanding, or uses of them.

New system levels bring new phenomena & DSLs

- As higher level system domains (e.g., aircraft engines, whole aircraft, multiple aircraft-satellites-ground stations) are modeled, larger scale, higher-level interactions appear, and from these emerge new semantic patterns, including domain specific languages (DSLs).
- This is exactly identical to the emergence of chemistry from atomic physics, and the emergence of biology from chemistry.
- It is not true that establishing lower-level semantic interoperability solves the interoperability of the higher-level domains.
- A similar misunderstanding in physics led to a famous paper ("More Is Different") by Nobel Laureate P. W. Anderson.
- This means we are never "done" creating semantic interoperability! The ability to do so must be practiced as an ongoing part of innovation.

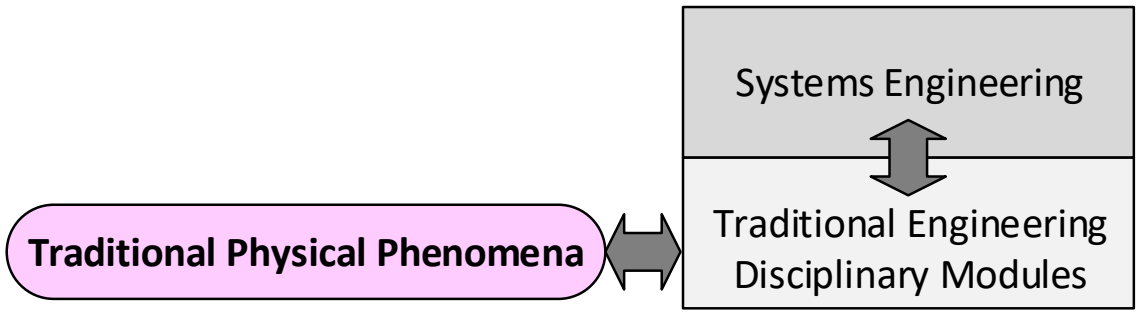
https://www.omgwiki.org/MBSE/lib/exe/fetch.php?media=mbse:patterns:science_math_foundations_for_systems_and_systems_engineering--1_hr_awareness_v2.3.2a.pdf

- The System Phenomenon and its supporting mathematics (Hamilton et al) provide the inductive ladder, explaining (*) theory of each new level in terms of the previous level.
- As higher-level system patterns are discovered, represented, validated, taught, and practiced, they become "emergent domain disciplinary frameworks".
- This is evident in the history of scientific and engineering domains and disciplines, and newer emerging ones.

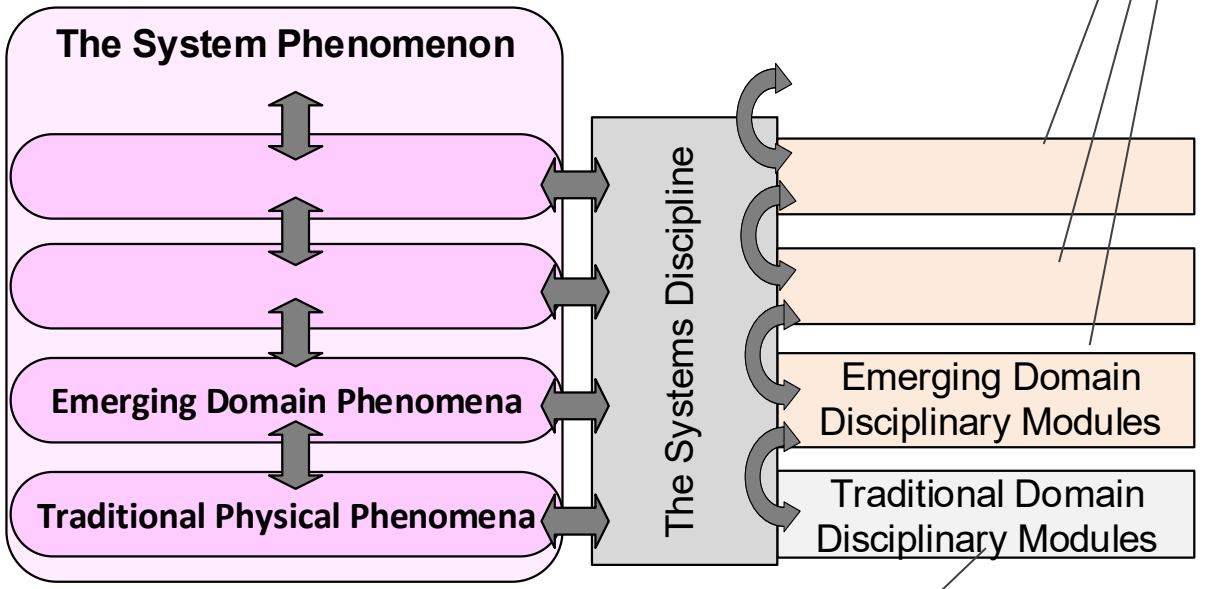
Future
- Distribution networks
- Biological organisms, ecologies
- Market systems and economies
- Health care delivery
- Systems of conflict
- Systems of innovation

Recent
- Ground Vehicles
- Aircraft
- Marine Vessels
- Biological Regulatory Networks

Traditional view:

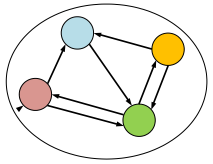


Future view:



ME, CE, EE, ChE, …

* Explaining after their discovery, but generally not predicting them before. See P. W. Anderson, 1972.



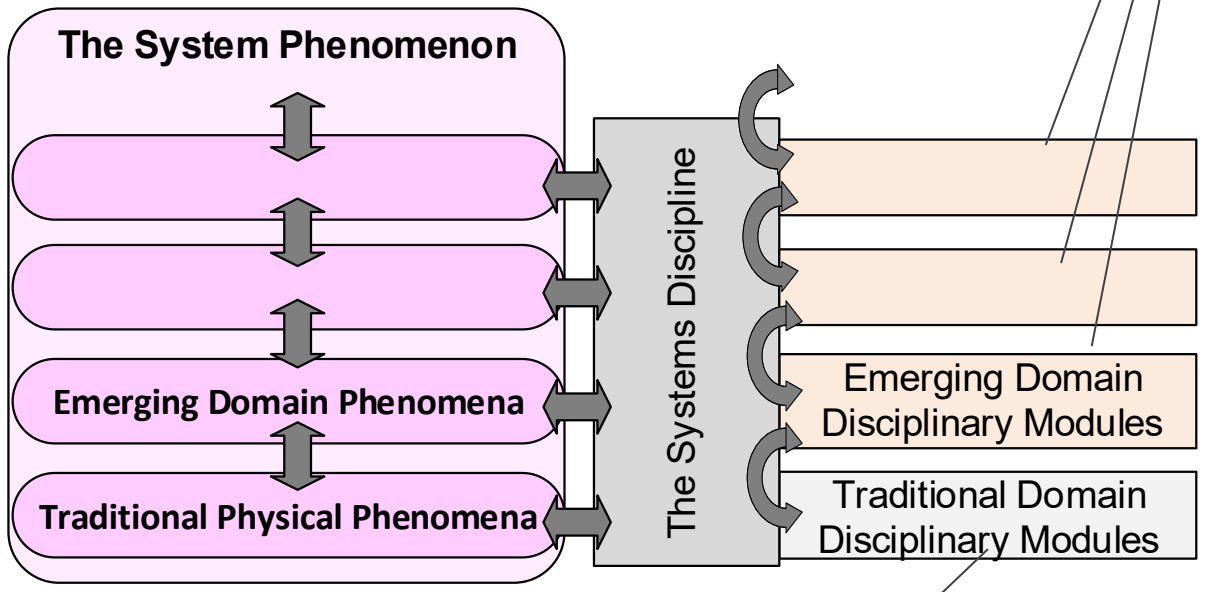
Impacts on *Semantic Structure* Emerge Uniquely for Each Emergent Domain

- New interactions (e.g., on the Internet) lead to new domains—each with new structure, new named things (roles), attributes, and relationships.
- Each new domain arising from new interactions thus creates a new ontology (domain specific language).
- So, a single "master ontology" is thus never enough!
- Domain ontologies are about semantic structure, not about quantitative mathematical aspects.
- Human skills and tools for language and meaning are called into play—different than quantitative skills and tools. Calls upon System Thinking.
- The related ontology frameworks thus have both structural semantic and quantitative math aspects.
- Here designers face a different "reverse" problem than scientists: Seeking to discover structure to produce interaction behaviors to deliver benefits (next section).

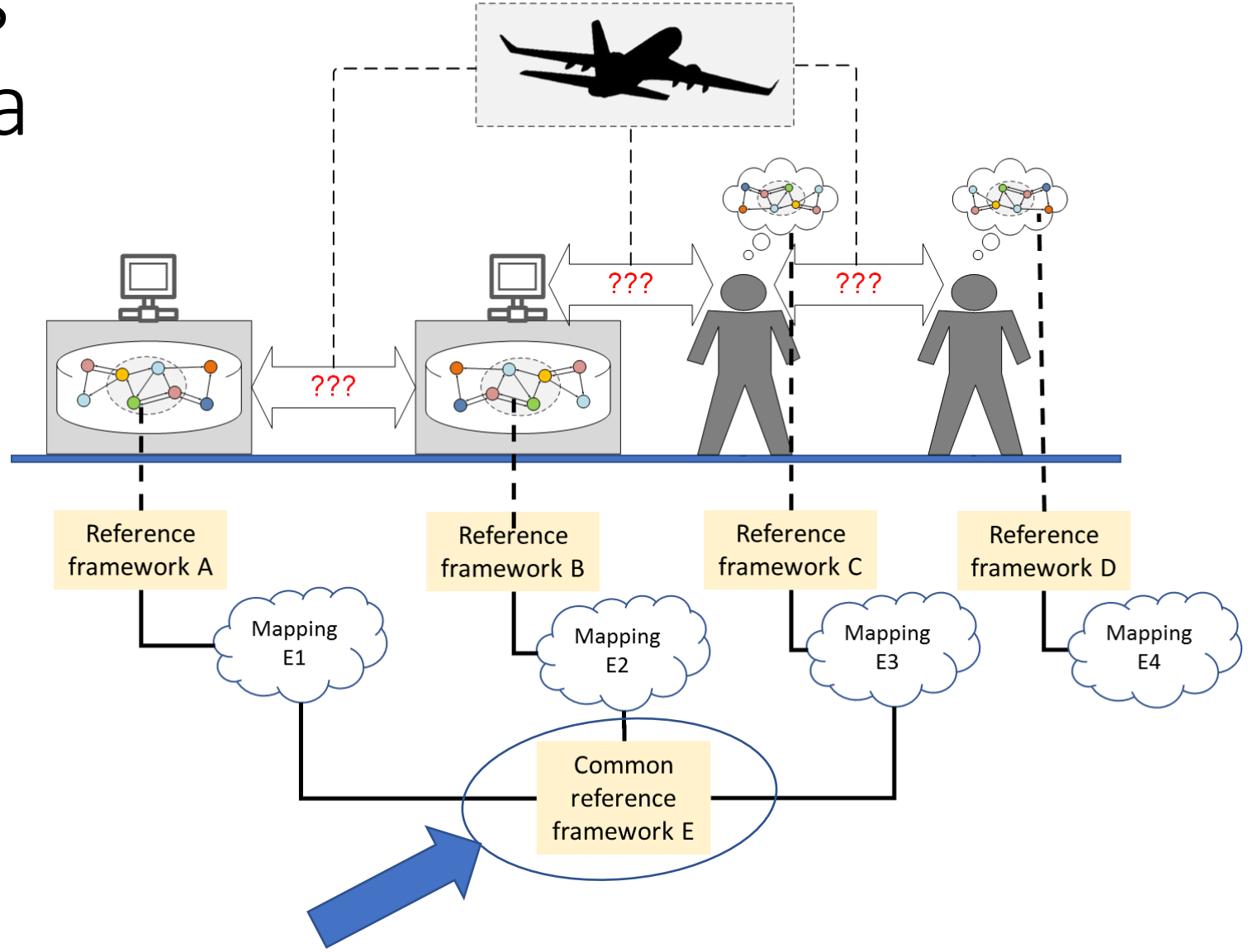
Future
- Distribution networks
- Biological organisms, ecologies
- Market systems and economies
- Health care delivery
- Systems of conflict
- Systems of innovation

Recent
- Ground Vehicles
- Aircraft
- Marine Vessels
- Biological Regulatory Networks

Future view:



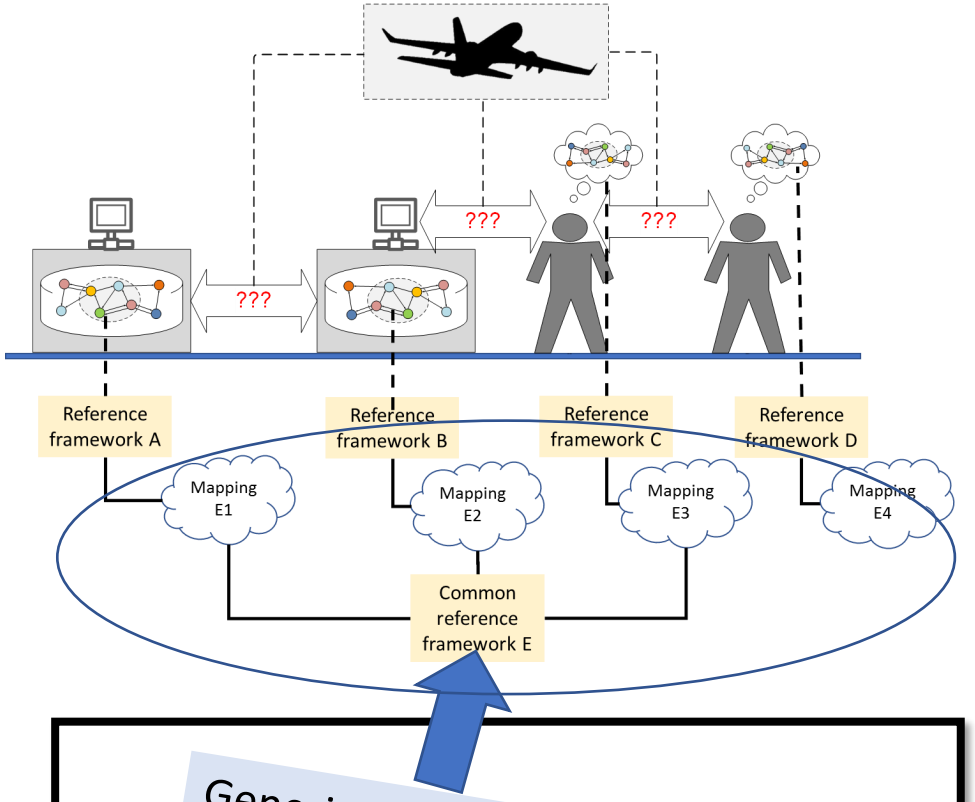
New interaction levels bring new phenomena and DSLs



Becomes a hierarchy of reference patterns, from general systems to intermediate patterns (e.g., cybersecurity) to specific domains (flight, medical, automotive, manufacturing, etc.)

What you can do right now (and should do first anyway)

- Use mappings of your tools and information systems to a common reference framework
- This does not require buying new tools, writing new programs, etc.
- It does require humans to work together—after all, "playing well together" is partly about people!
- This establishes the basis of semantic interoperability, without getting into other automation issues until you are ready to do so later.
- Example: Computational Model – System Model "phenomena" mapping demonstrated in V4I Projects by System Context Models.
- You can do this now, if you have the will; if you don't, then more programmers and software packages won't help much later.
- Map to S*Metamodel, domain specific S*Patterns, etc.



Generic metamodel mapping-- the first level mapping.

S*Metamodel Mapping

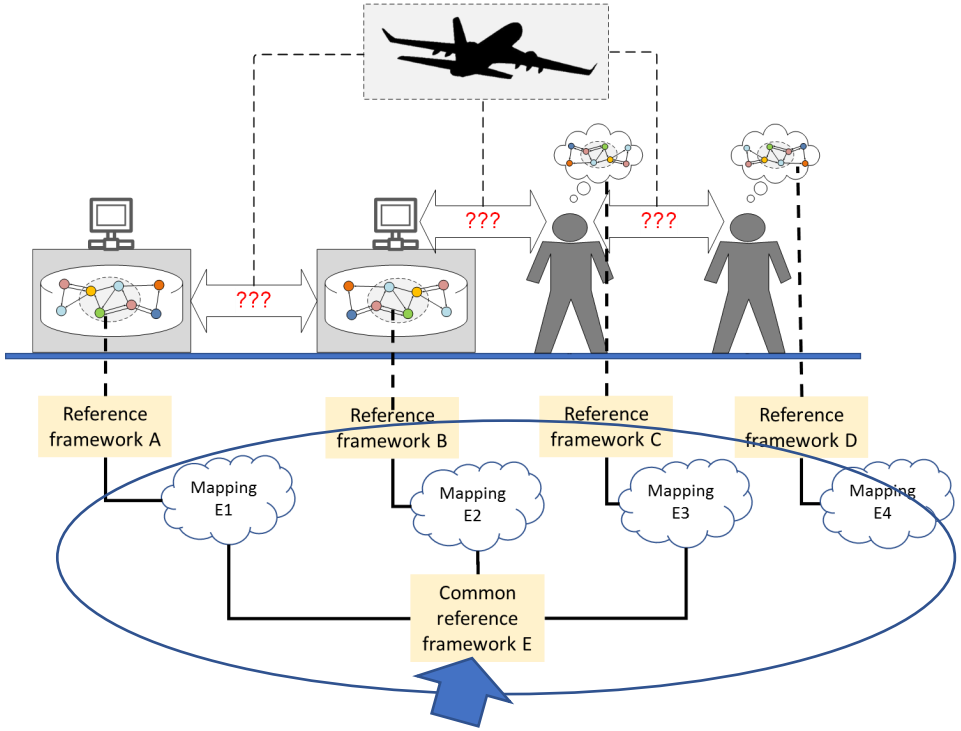
for

Cameo Systems Modeler

Versions 2019-2024xR2

Group learning: Essential for "playing well together"

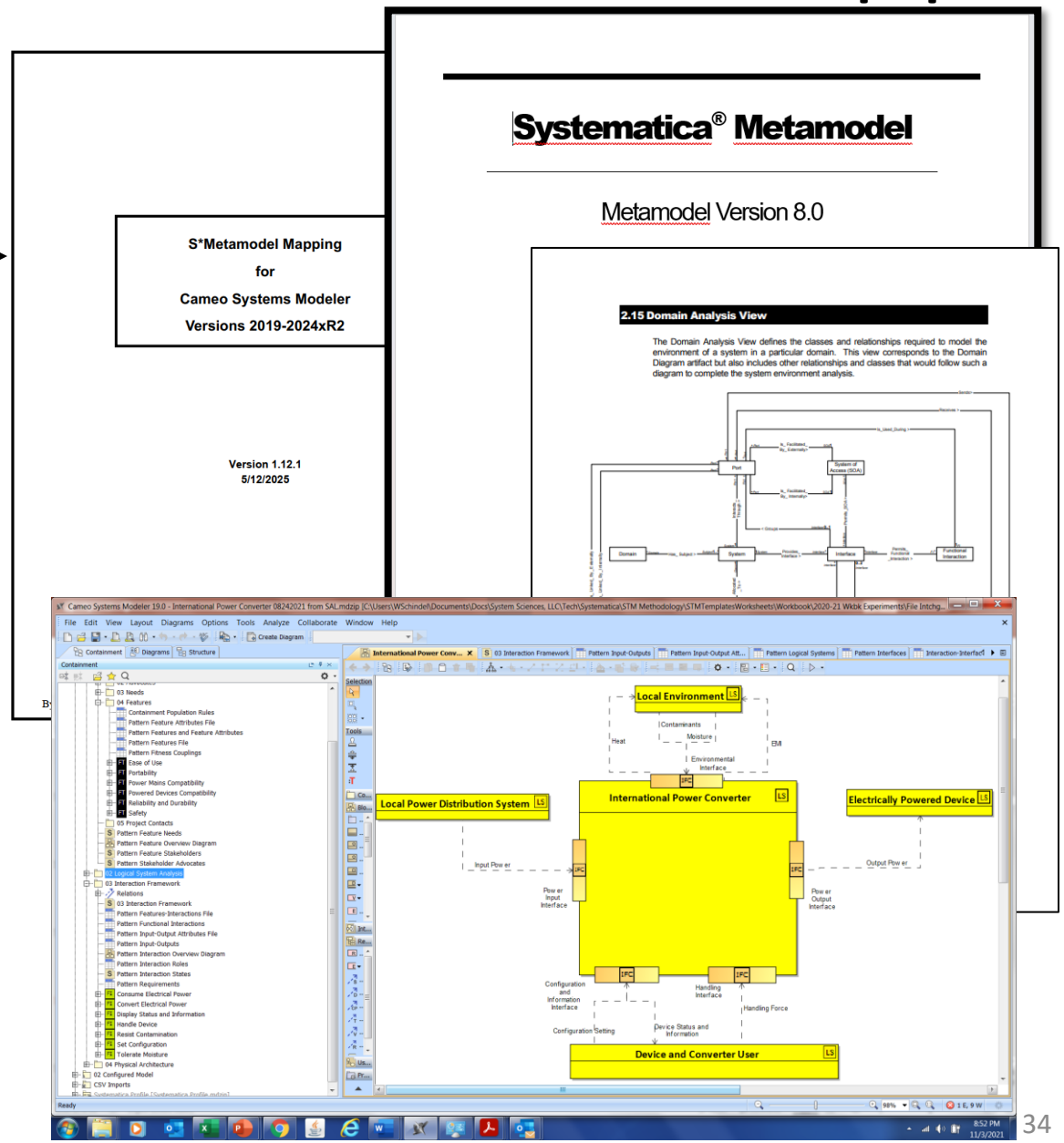
- The higher-level domain mappings are in effect discovery of recurring general patterns that can be "configured" for individual cases.
- This is what we can engage in together during these class sessions.
- In an innovation organization, Group Learning is the essential core of "Playing Well Together".
- Uncover the Pattern™ (UTP) is a fast way to make this happen for a specific domain pattern.



Domain pattern mappings are the specialized levels of mapping.

Examining the Form of a Formal Mapping

- Specifically, the Formal S* Metamodel Mapping for MagicDraw/Cameo Systems Modeler Versions 2019-2024xR2.
- Available on the shared site and the reference links.



Examining the Form of a Formal Mapping

- High-Level Summary Table Extract from Mapping document showing a subset of S* Metamodel elements mapped into SysML elements.
- Let's look at the Mapping document more closely together.

Mapping Letter	Systematica Meta-Model Element	SysML Element Type
A	Stakeholder	Block with «Stakeholder» stereotype
B	Feature	Block with «Feature» stereotype
C	Functional Interaction (Interaction)	Activity with «Functional Interaction» stereotype
D	State	State with «State» stereotype
E	System	Block with «Logical System» stereotype
F	Interface	Proxy Port with «ifc port» stereotype typed by an Interface Block with an «Interface Definition» stereotype
G	System of Access	Block with «System of Access» stereotype
H	Input/Output	itemFlow that conveys a Classifier with an «Input_Output Definition» stereotype
I	Fitness (A) Couplings	Constraint Block with «Fitness Coupling» stereotype
J	Functional Role	Block with «Logical System» stereotype being referenced by an Interaction via a shared aggregation relationship with «Has Role» stereotype.
K	Requirement Statement	Requirement with «Requirement Statement» stereotype
L	Design Component	Block with «Physical System» stereotype
M	Characterization (B) Couplings	Constraint Block with «Characterization Coupling» stereotype
N	Feature Attribute	Attribute with «Feature Primary Key» or «Feature Attribute» stereotypes
O	Role Attribute	Attribute with «Logical System Attribute» stereotype
P	Design Component Attribute	Attribute with «Physical System Attribute» stereotype
R	Decomposition (C) Matrix Couplings	Constraint Block with «Decomposition Coupling» stereotype
S	IO Attribute	Attribute with «IO Attribute» stereotype
T	Input-Output (D) Coupling	Constraint Block with «Input Output Coupling» stereotype

Examining the Form of a Formal Mapping

- One section describes all of the S* Metamodel elements mapped into SysML elements.
- Let's look at the Mapping document more closely together.

4 Detail Mapping

*4.1 S*Metaclass Mapping Details*

The Systematica Class mappings to SysML is detailed by the following table:

Systematica Modeled Class Name	SysML Element
Allowed Value	Block with «Allowed Value» stereotype
Architectural Relationship	Simple: Name of the Association relationship between systems with «architectural relationship» stereotype; Reified: Block with «Architectural Relationship Definition» stereotype
Architectural Relationship Role	Simple: Association Target and Source Roles; Reified: Dependency with «Has AR Role» stereotype
Attribute Coupling	Constraint Block with «Attribute Coupling» stereotype
Attribute Coupling Map	Constraint of a Constraint Block, Artifact referenced or linked to by Constraint
Attribute Role	Parameter of Constraint Blocks
Class	This is a superclass of other Metaclasses and is not mapped.
Counter Requirement	Requirement with «Counter Requirement» stereotype
Design Component	Block with «Design Component» stereotype
Design Component Attribute Role	Parameter of Constraint Blocks
Design Constraint	Requirement with «Design Constraint» stereotype
Design Constraint Statement	Requirement with «Design Constraint» stereotype
Design Coupling	Constraint Block with «Characterization Coupling» stereotype.
Design Coupling Map	Constraint within Constraint Block, Artifact referenced or linked to by Constraint
Domain	Block with «Domain» stereotype
Domain System	Block with «Logical System» stereotype aggregated within a Domain
Event	Trigger with «Event» stereotype
Failure Impact	Block with «Failure Impact» stereotype
Failure Mode	State with «Failure Mode» stereotype
Failure Mode Context Element	Block with «Failure Mode Context Element» stereotype
Feature	Block with «Feature» stereotype
Feature Attribute Role	Parameters of Constraint Blocks
Functional Interaction	Activity Block with «Functional Interaction» stereotype.
Functional Role	Block with «Logical System» stereotype and in Aggregation relationship with «Has Role» stereotype with Interaction

Examining the Form of a Formal Mapping

- One section describes all of the S* Metamodel relationships mapped into SysML elements.
- Let's look at the Mapping document more closely together.

4.2 S*Metarelationship Mapping Details

The S*Metarelationship mappings to SysML can be summarized by the following table:

Systematica Modeled Relationship Name	SysML Relationship	Source Class (mapped SysML element)	Target Class (mapped SysML element)
Advocates	Dependency with «Advocated_By (N-ADV)» stereotype	Logical System Block with «Advocate» stereotype that only participates in advocate-type relationships	Requirement with «Need» stereotype
Abnormal State Of	Dependency with «Abnormal State Of» stereotype	State with «Failure Mode» stereotype	Block with «Design Component» stereotype
Allocated To (Design Component-Functional Role)	Association with «Functional Role Allocation» stereotype for LS-DC	Block with «Logical System» stereotype	Block with «Design Component» stereotype
Allocated To (Modeled Attribute-Attribute Role)	Binding Connection for Attribute-Constraint Property or connection between Flow -Action Pin	Parameter of Constraint Blocks	Attribute with «Feature Attribute» or «Feature Primary Key» or «IO Attribute» or «Logical System Attribute» or «Physical System Attribute» stereotypes

Examining the Form of a Formal Mapping

- One section describes all of the S* Metamodel attributes mapped into SysML elements.
- Let's look at the Mapping document more closely together.

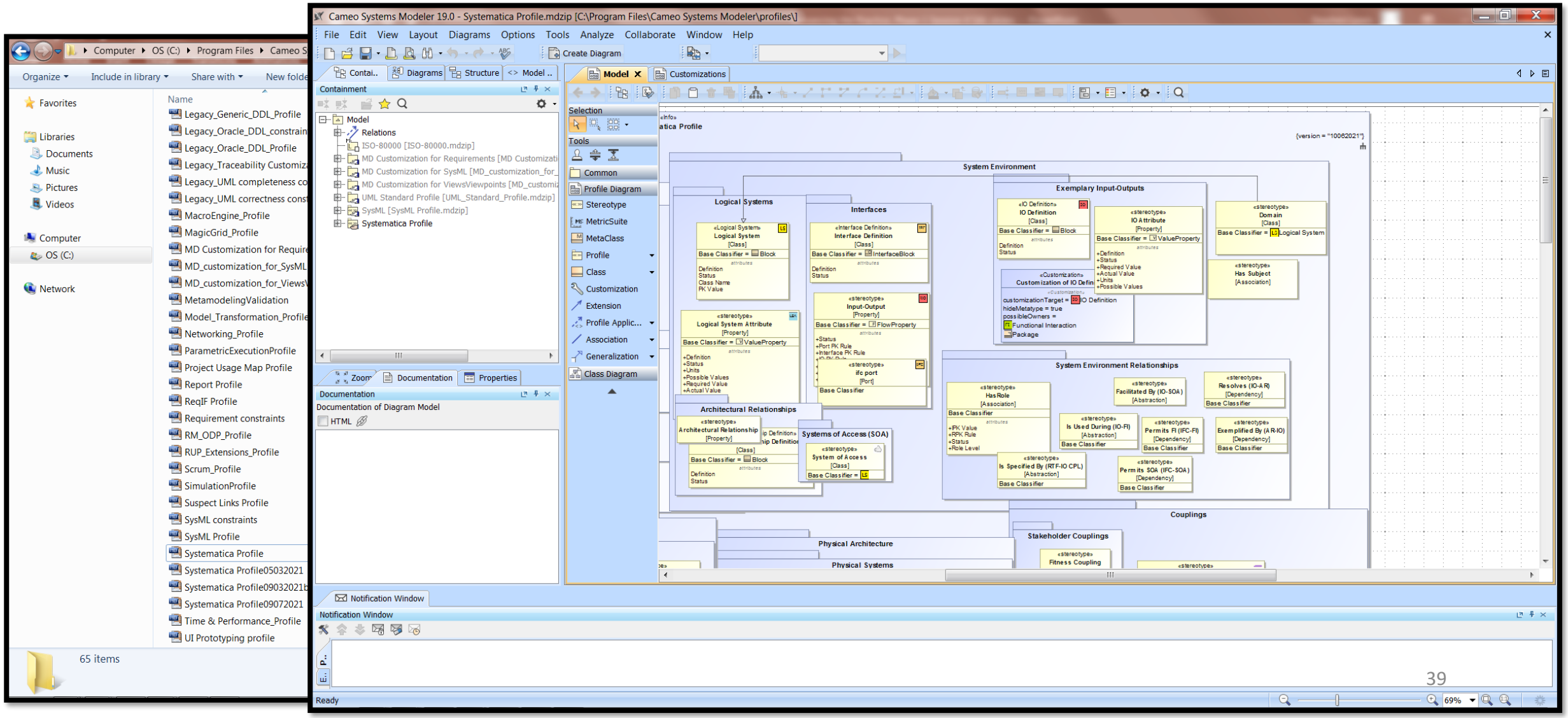
4.3 S* Attribute Mapping Details

This section of the mapping document maps Systematica Attributes of metaclasses and metarelationships, including specific pattern configuration rule attributes and common core attributes, into SysML properties, usually tags. Additionally, this section indicates which SysML element owns those properties. This section does not map the Systematica element to the SysML element. The details of those mappings are in Sections 1 and 2 and may differ from what is shown here, when an attribute is not mapped to the same owner that its metamodel owner is mapped to.

Systematica Attributes	Attribute Owner Systematica Metamodel Element	SysML Properties	Property Owner SysML Element
Interaction Population Rule	Uses Functional Interaction (Relationship)	"FPK Value" tag	Aggregation with «Uses Functional Interaction» stereotype
Interaction PK Value Rule		"IPK Rule" tag	
Role Population Rule	Has Role (Relationship)	"IPK Value" tag	Aggregation with «Has Role» stereotype
Role PK Value Rule		"RPK Rule" tag	
Design Component Population Rule	Allocated To (Relationship)	"Configuration Rule" tag	Aggregation with «Functional Role Allocation» stereotype
Design Component PK Value Rule		"IPPK Value" tag	
Requirement Population Rule-Interaction	Requirement Transfer Function (Class)	"IPK Rule" tag	Class with «Requirement Statement» stereotype
Requirement Population Rule-Role		"RPK Rule" tag	
Requirement Statement PK Value Rule		"RSPK Rule" tag	
State Population Rule-Interaction	Requires (Relationship)	"IPK" tag	Abstraction with «Requires» stereotype
State Population		"RPK" tag	

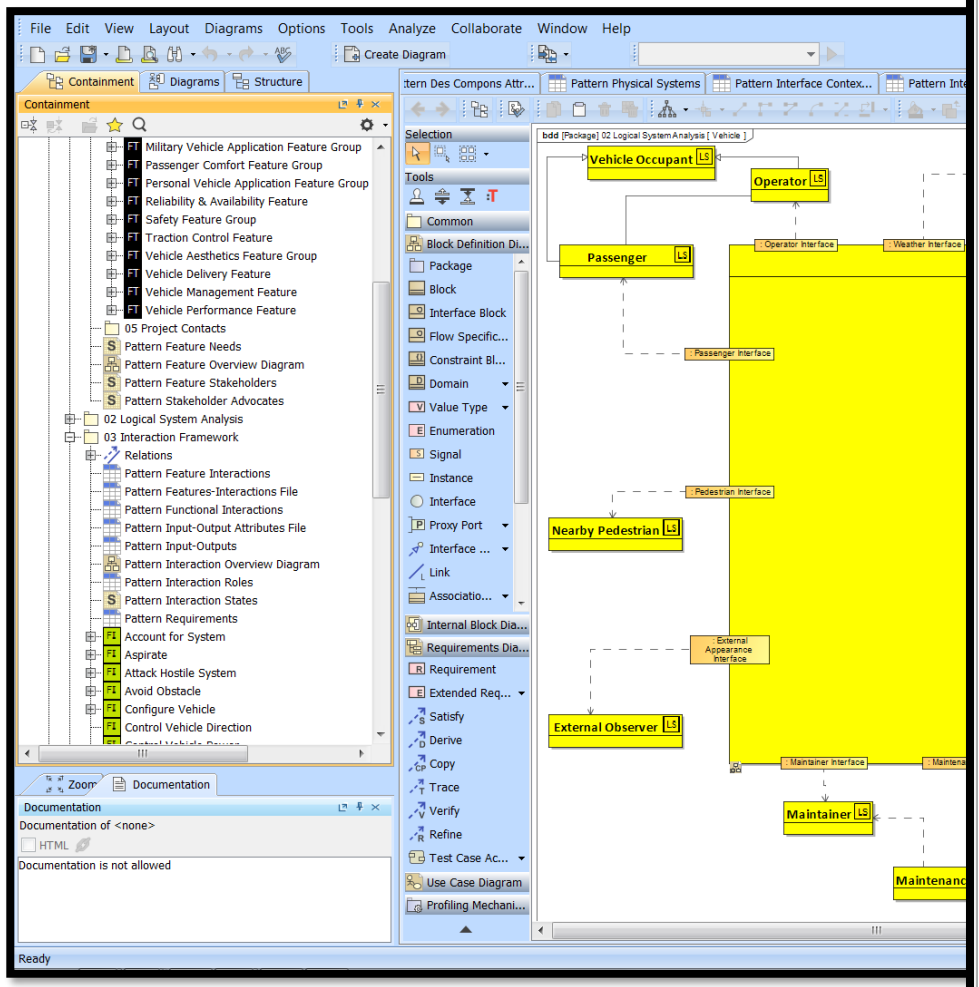
A Tool Specific Profile; Installing and Navigating

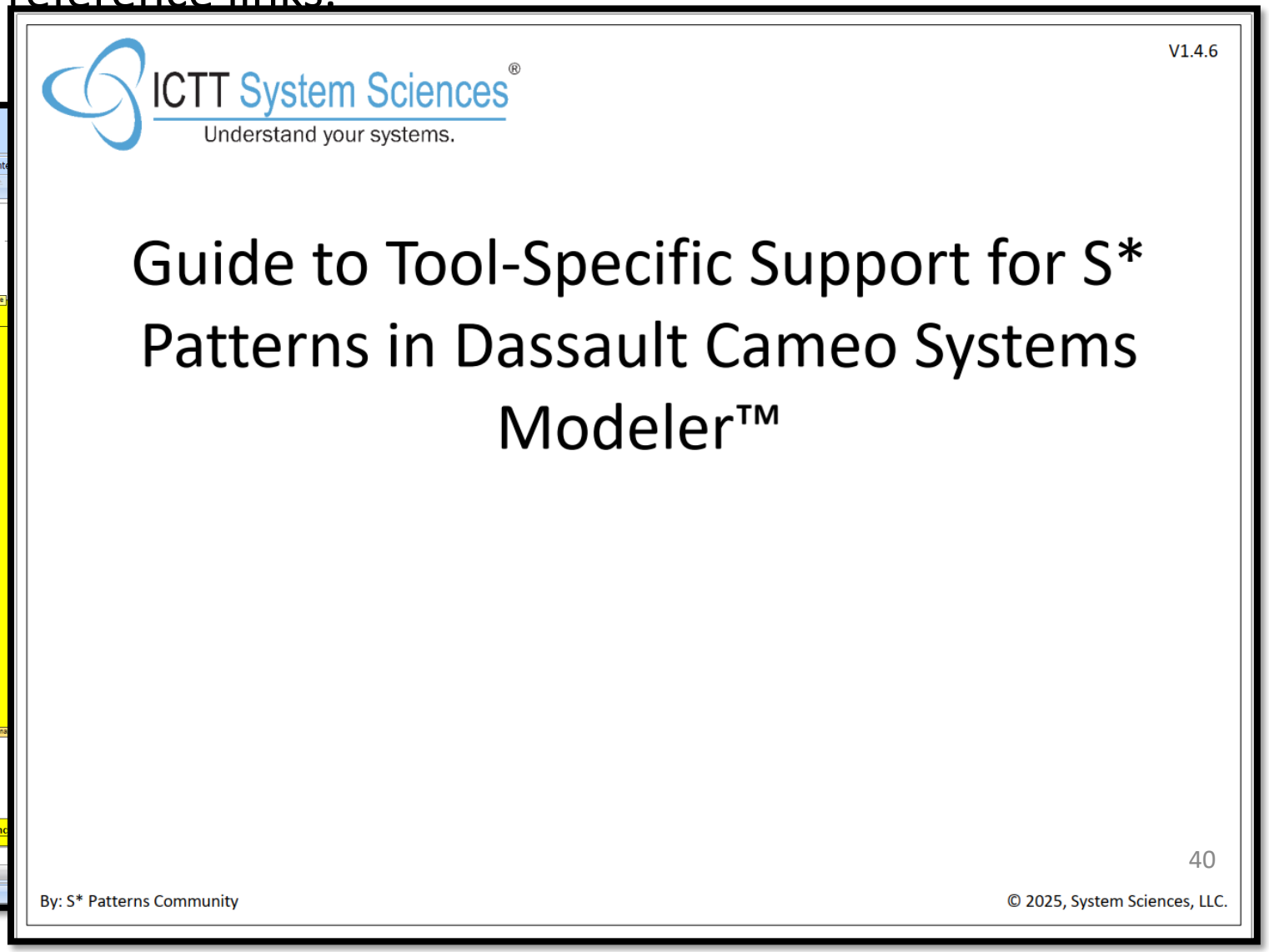
- Specifically, the Systematica Profile for Cameo Systems Modeler.
- Available on the shared site and the reference links.



A Tool Specific Profile; Installing and Navigating

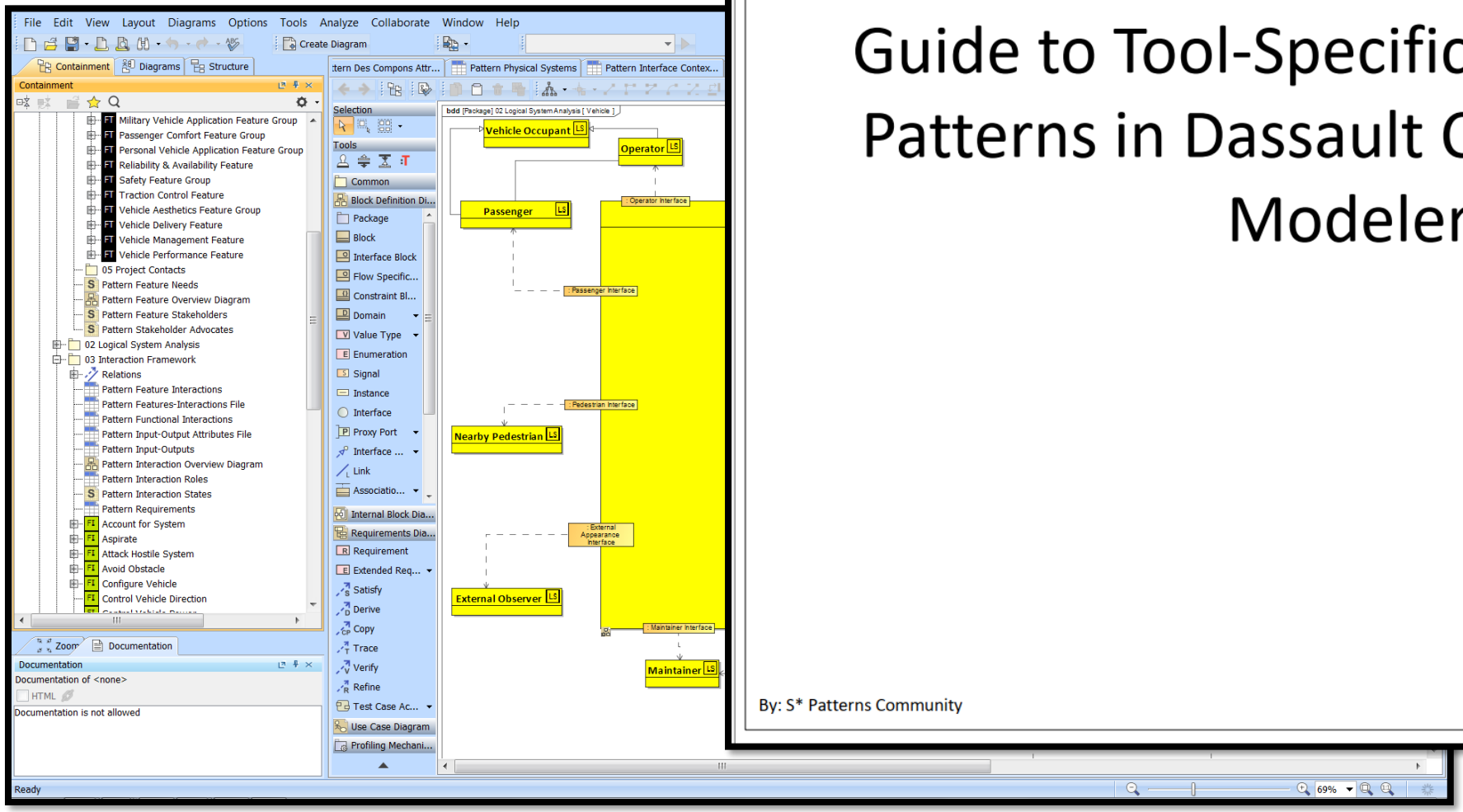
- Let's look at the Profile Installation Guidance documentation and Pattern in CSM together.
- Available on the shared site and the reference links.

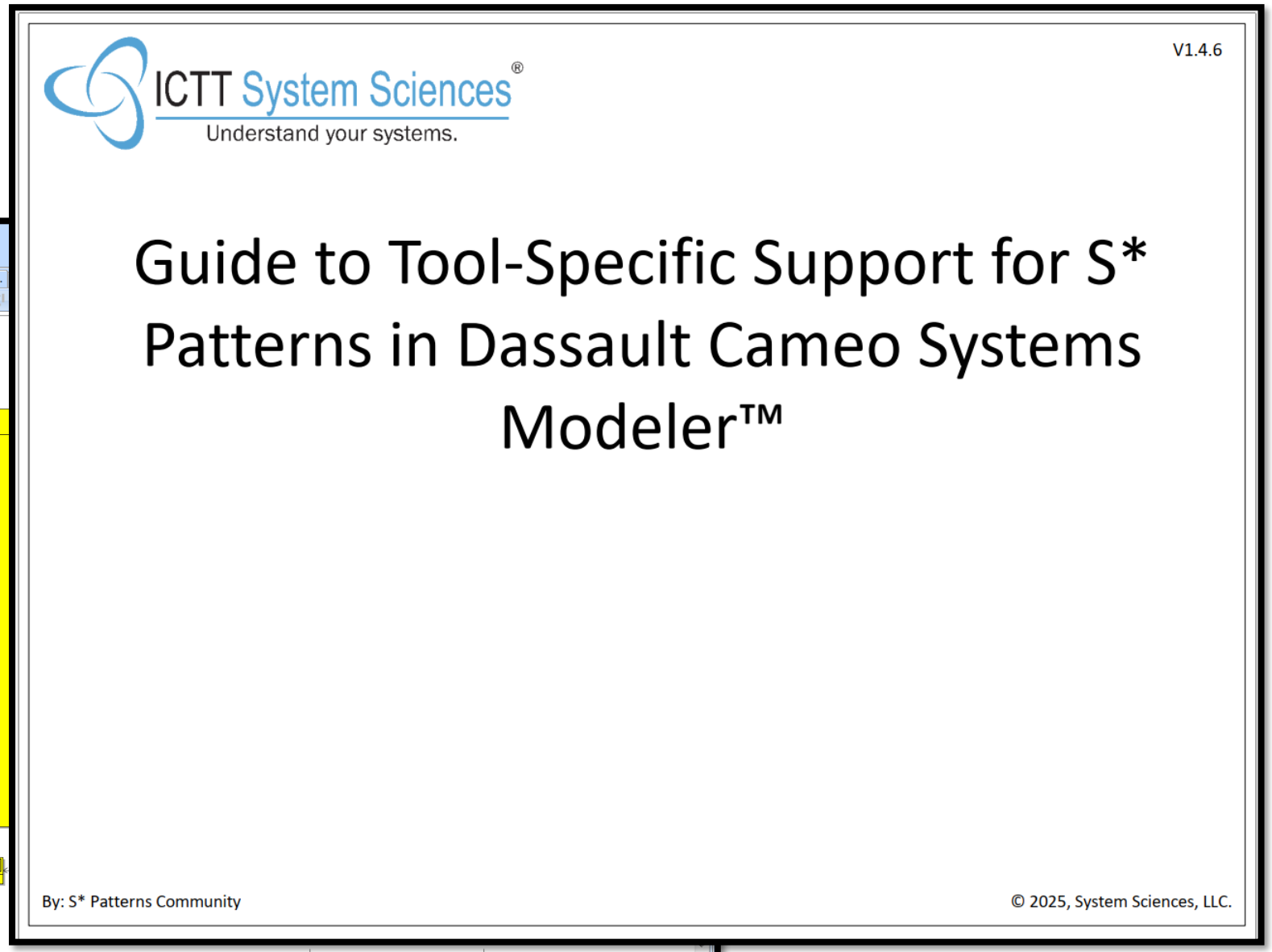




A Tool Specific Profile; Guide to Entering a Pattern

- Use the Guide to help enter Pattern details.
- Available on the shared site and the reference links.





Semantic technologies; OWL; ST4SE project

- Significant global investment has been made over the last two decades in “semantic technologies”:
  - Tools, languages, repositories, standards for information technologies focused on increasing the value of information encoded with stronger semantics.
  - e.g., W3C Web Ontology Language (OWL), authoring tools (Protégé), reasoning tools (Pellet), query language (SPARQL), and triplestores.
  - A significant part of this is Semantic Web Technologies—those meant to leverage the value of web-based information encoded in the related standards of W3C.
  - There are also additional semantic technology developments outside the Semantic Web.
  - SysML V2.0 (based on a new metamodel!) is slated to provide OWL compatibility.

Other tools and semantic frameworks; semantic technologies; OWL; ST4SE project

- Somewhat outside the scope of this class, but interested participants can consult the Semantic Technologies for Systems Engineering (ST4SE) Project of the INCOSE MBSE Patterns Working Group . . .
- Demonstrating both model configuration (generation) and model checking using the same S*Pattern.

https://www.omgwiki.org/MBSE/lib/exe/fetch.php?media=mbse:patterns:incose_patterns_wg_report--st4se_project--1.10.6.pdf

- A major promise of Semantic Technologies for the Systems Engineering community is advancing automation of Consistency Management, which is the heart of the Ecosystem Pattern we will be discussing in the Phase II classes of this series.

In-class Exercises

- What is the purpose of the S*Metamodel?
- What is the purpose of an S*Metamodel Mapping?
- What is the difference between an S*Profile and an S*Mapping for a tool?
- What is the purpose of the “Guide to Tool Specific Support of S*Patterns in Dassault Cameo Systems Modeler”?
- In the diagram on Page 7 of these class slides…
  - Find your own role(s)
  - Find roles of others in your company or supply chain
  - Find the S*Metamodel
  - Find an S*Mapping
  - Find an S*Profile

Homework assignment 4

- In the table of contents of the S*Metamodel Document, what is the purpose of each major section?
- In the table of contents of the “S*Metamodel Mapping for MagicDraw/Cameo Systems Modeler Versions 2019-2024xR2”, what is the purpose of each major section?
- In the table of contents of the “Guide to Tool Specific Support of S*Patterns in Dassault Cameo Systems Modeler”, what is the purpose of each major section?
- In the S*Metamodel Document, find the metamodel segment diagrams for States (Section 2.7), Risk Analysis (Section 2.24), and Features (Section 2.4). Why does Functional Interaction appear in all three?
- If models are created by two modelers using the same modeling language and tools, why won't they automatically be compatible with each other?

Discussion, Q&A.

-
-
-
-
-
-



References

1. “Formal S*Metamodel”, https://www.omgwiki.org/MBSE/lib/exe/fetch.php?media=mbse:patterns:systematica_8_metamodel_v8.0.61.pdf
2. “S*Metamodel Mapping for MagicDraw/Cameo Systems Modeler”, Version 19 https://www.omgwiki.org/MBSE/lib/exe/fetch.php?media=mbse:patterns:systematica_mapping_for_magicdraw_csm_v1.12.1.pdf
3. “What Is the Smallest Model of a System?”, https://www.omgwiki.org/MBSE/lib/exe/fetch.php?media=mbse:patterns:what_is_the_smallest_model_of_a_system_v1.4.4.pdf
4. “Maps Versus Itineraries: A Systems Engineering Lesson from Ancient Mariners”, https://www.omgwiki.org/MBSE/lib/exe/fetch.php?media=mbse:patterns:maps_vs._itineraries_v1.3.1.pdf
5. “Guide to Systematica Patterns in Cameo Systems Modeler”: https://www.omgwiki.org/MBSE/lib/exe/fetch.php?media=mbse:patterns:guide_to_systematica_patterns_in_cameo_systems_modeler_v1.4.6.pdf