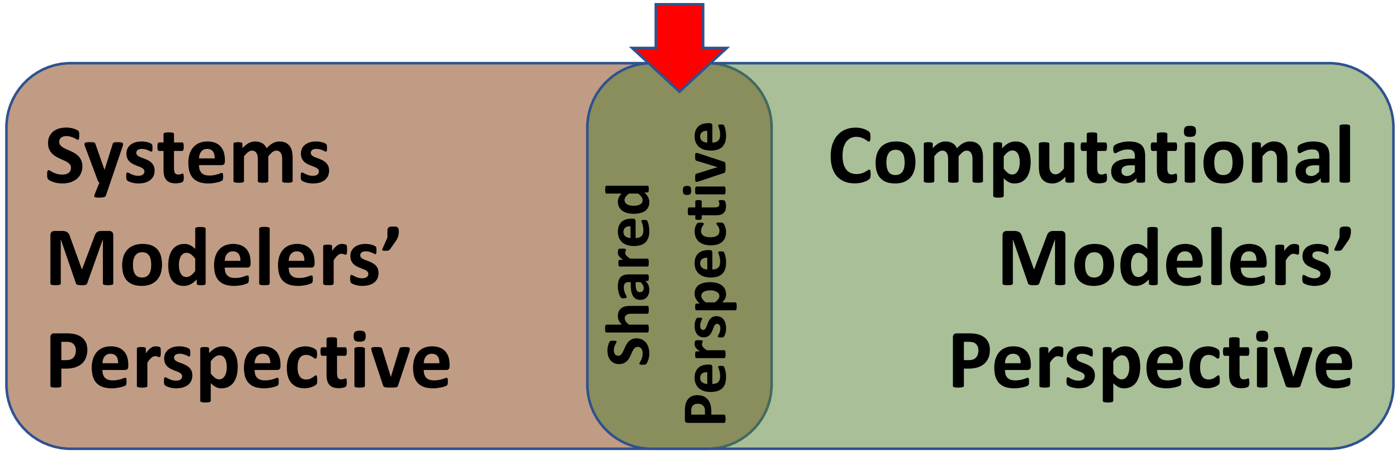
Discipline-Specific Computational Modelers *vs.* Systems Modelers:

Identifying A Shared Perspective and Organizational Interface



ICTT System Sciences®
Understand your systems.

V1.3.1

The Challenge of Two Different Perspectives

Computational Modelers' Perspective

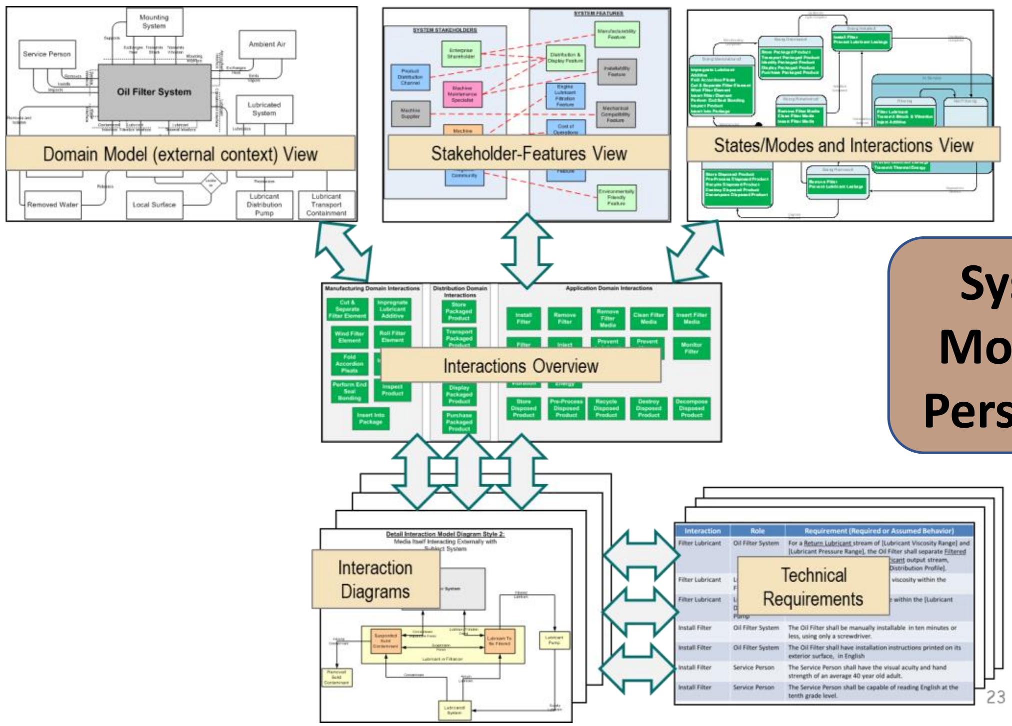
- Discipline-specific computational modelers' perspectives often emphasize:
  1. Dynamical behavior frequently represented by time-based simulation of that behavior
  2. Optimization or analysis at the level of the simulated phenomena
  3. Numerical methods, simulation platforms, simulation languages
  4. Physical sciences (for physics-based models), or . . .
  5. Machine learning (for data-driven models)
  6. High performance computing

Systems Modelers' Perspective

- System modelers' perspectives often emphasize:
  1. Multi-level organizing frameworks of collective structure, behavior, interfaces
  2. Architectures, partitioning (decomposition) and hierarchical viewpoints
  3. Global, emergent system behavior represented, not necessarily by dynamical simulation
  4. All dimensions represented, not just behavior, multiple views, product lines and families
  5. Systems modeling languages, tools, platforms

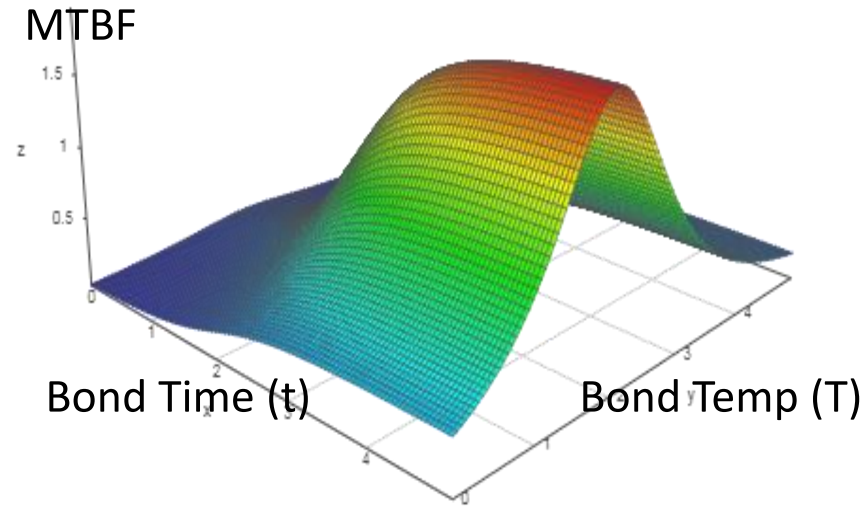
Challenge of different human perspectives is not just a matter of integrating different models

- Technical standards such as FMI can help greatly in integrating different technical models, but . . .
- The challenge discussed here is the very different <u>human</u> perspectives of <u>systems</u> versus <u>discipline-specific</u> engineers



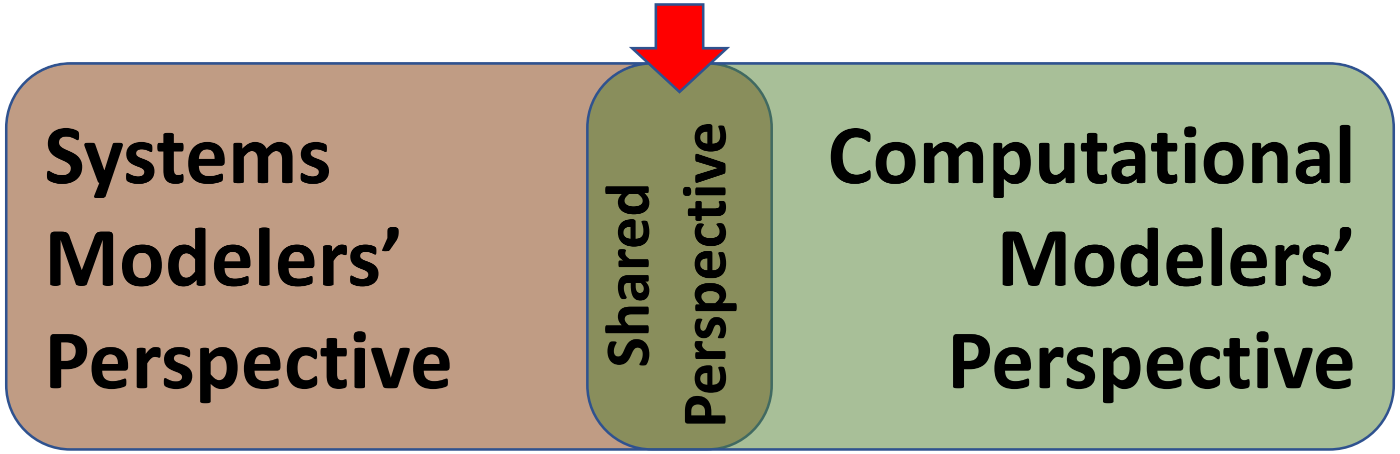
Systems Modelers' Perspective

Computational Modelers' Perspective

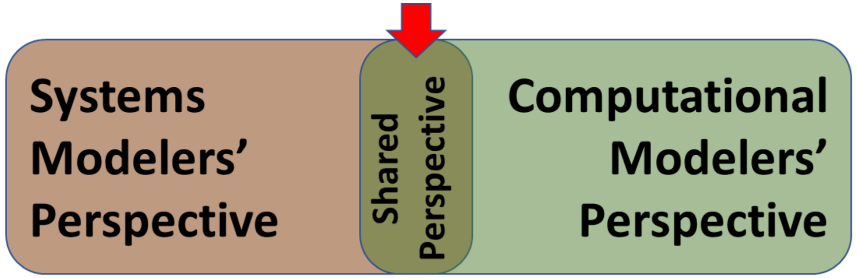


Good news: There is a shared perspective

- It is <u>not necessary</u> to demand that these skilled specialists fully adopt each others’ overall skillsets and perspectives.
- Instead, making a few simple agreements about a <u>shared</u> (overlapping) part of the two perspectives can be sufficient to <u>align the work of two groups</u>.



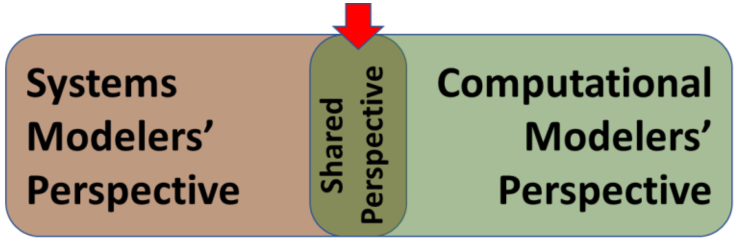
The three elements of the Shared Perspective



Practitioners can align by agreeing on the respectively modeled:

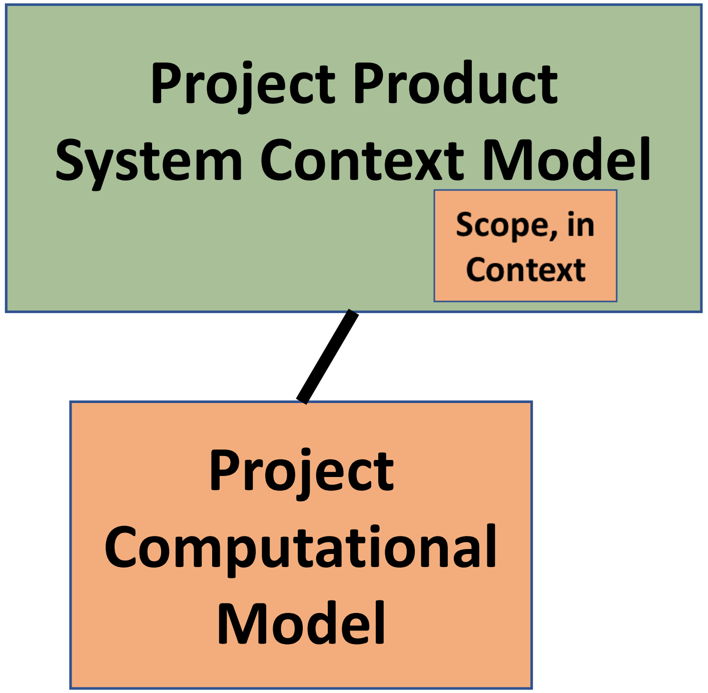
1. Phenomena / Interactions--
   - Examples: Combustion; Melting; Corrosion.
2. Variables / Attributes / Parameters--
   - Examples: Temperature; Efficiency; Material Type; Tensile Strength; Life.
3. Attribute Couplings / Dependencies / Equations / Laws--
   - Examples: Elasticity Stress-Strain Curve; Purity-Strength Correlation; Equations of Motion; Material Properties Table; Customers' Product Selection Guide.

Actions to take



- Computational Modelers: Identify the following, as would already be expected in computational modeling work--
  - Phenomena, ranked by priority (PIRT)
  - Attributes (variables, parameters) for the computational model
  - Attribute couplings (dependencies, equations, laws) to be simulated (even if to be machine learned)
- Systems Modelers:
  - Map the Attributes to identified system Input-Outputs, Functional Roles (with their Functional Interactions), Stakeholder Features, and Design Components.
  - Preferably, choose those Input-Outputs, Roles, Interactions, Features and Design Component from standard System Patterns (S*Patterns) for the enterprise or domain.
  - Show the resulting system model to Computational Modelers, for their confirmation of the system context of their work.
- For both groups:
  - As needed, negotiate the above to consistency, including shared naming necessary for alignment
  - Note that any view format (table, diagram, list, otherwise) can accomplish the above

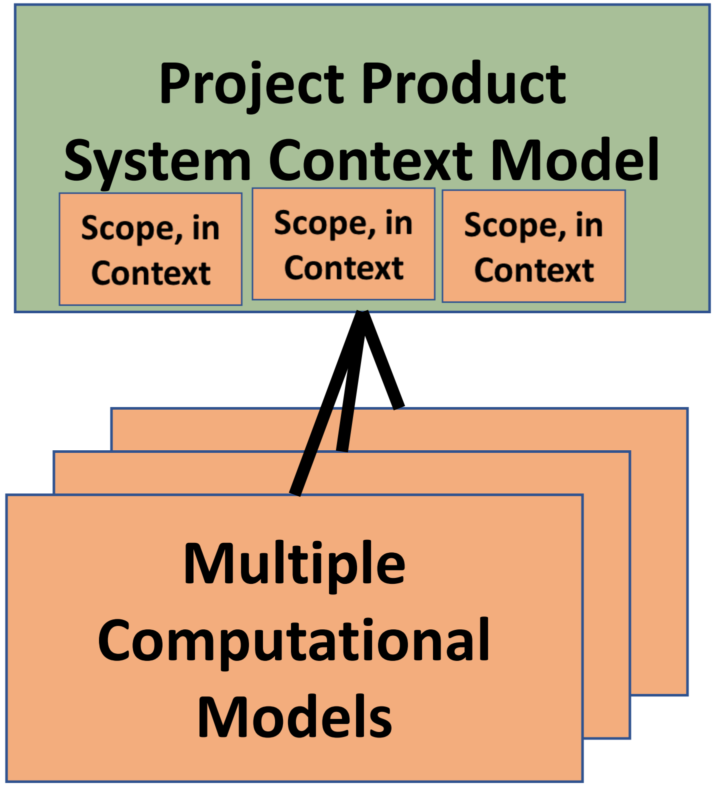
Payoffs for acting: Who benefits, and how?



1. Benefits in a single project, single computational model:
   - The systems context model, with the computational model embedded in it by the above actions, makes explicit (to both groups) the relative scope and context of the computational model, within the larger system context.
   - This often leads to insights or event surprises as to the scope or context—at the very least, a common view of that context.
   - Who gains / realizes the benefit? Both the computational modelers and the larger system context / business may gain from this discovery, earlier than might otherwise been the case. It may lead to earlier adjustment of or recognition of value of the computational model.
   - Example: The teams realize that the scope of the computational model does not include some scope that had been assumed by its name or prose description, or includes more scope than had been realized.

(more)

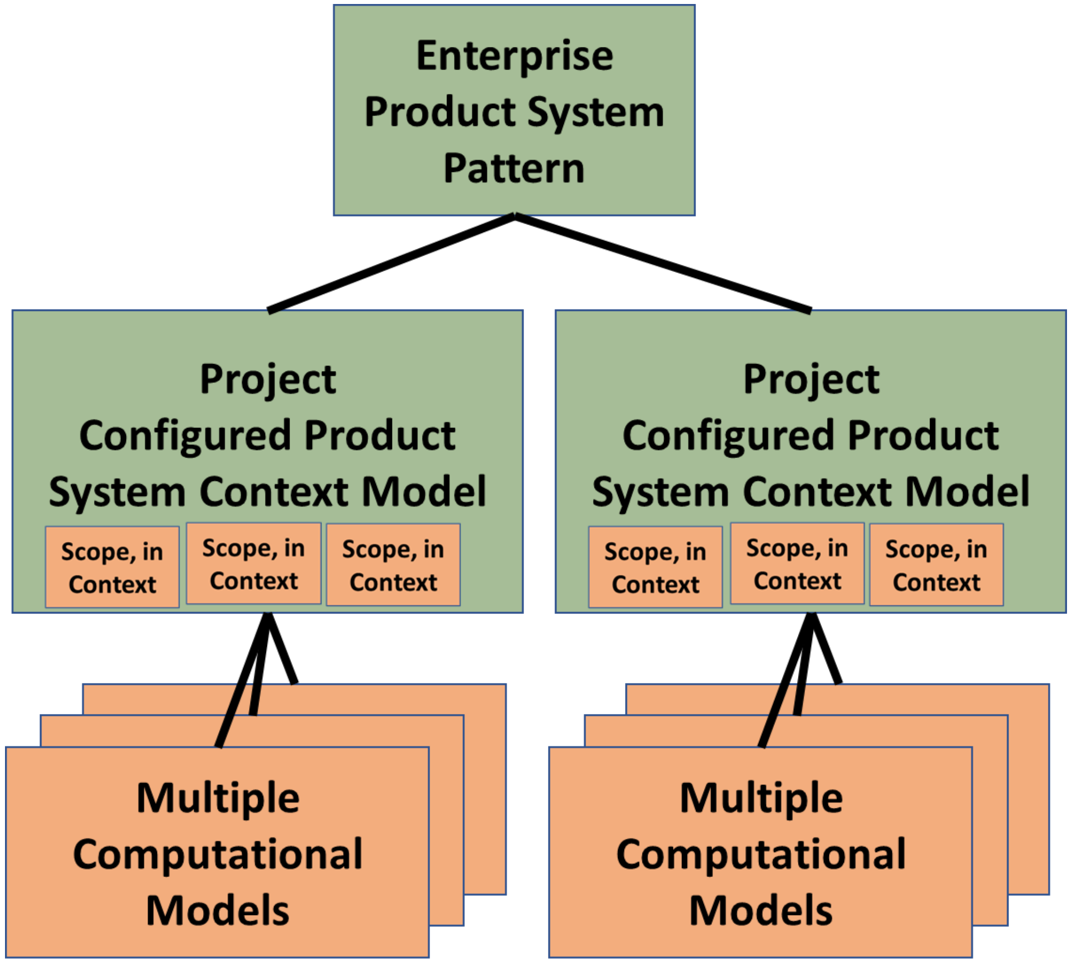
Payoffs for acting: Who benefits, and how?



2. Benefits in a single project, multiple computational models:
   - The systems context model, with multiple computational models embedded in it by the above actions, makes explicit (to both groups) the relative scope and context of the full set of computational models, within the system context—including their relations to each other.
   - This often leads to insights or event surprises as to the scope of the full set and their relations—at the very least, a common view of that context.
   - Who gains / realizes the benefit? Both the computational modelers and the larger system context / business may gain from this discovery, earlier than might otherwise been the case. It may lead to earlier adjustment of or recognition of value of the computational models.
   - Example: The teams realize that the connection of two computational models does not include some scope that had been assumed by its name or prose description, or includes more scope than had been realized.

(more)

Payoffs for acting: Who benefits, and how?



3. Benefits across multiple, projects, programs, the enterprise:

- Across multiple projects, the use of a shared system product pattern leads to system level convention consistencies across multiple system projects and products.
- This benefit extends to computational models when they are aligned (by the method summarized earlier above) with the system context models.
- When it comes to VVUQ of the computational models, or trust in their model credibility, this can be leveraged to assess or gain that credibility on a leveraged basis.
- Who gains / realizes the benefit? The enterprise as a whole gains.
- Example: Two different product configurations are passing through their product development cycles. A shared computational model is used in both products/projects, with attention to its range of validation and verification.

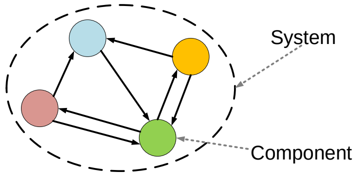
Especially for Systems Specialists: Additional Detail and Examples, Using S*Metamodel

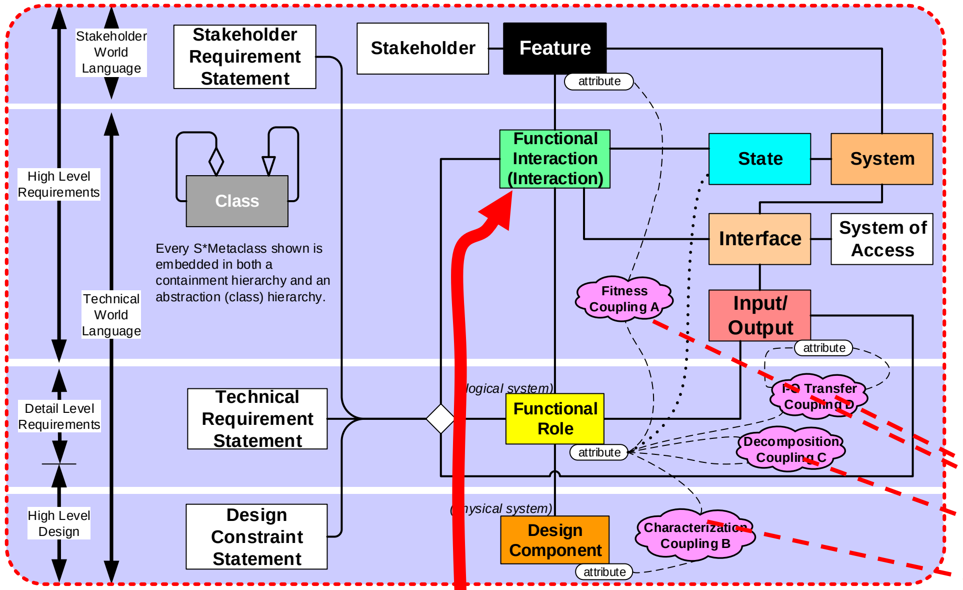
1. Phenomena / Interactions
2. Variables / Attributes / Parameters
3. Attribute Couplings / Dependencies / Equations / Laws

1. Phenomena occur in Context of Interactions.

Interactions occur between system components through the exchange of force, energy, material, or information, leading to changes of state.

Examples: Combustion, Melting, Corrosion

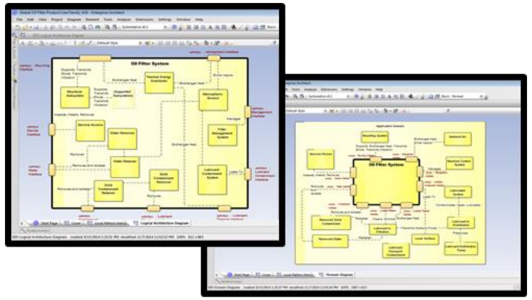




S*Metamodel informal summary pedagogical diagram
(formal S*Metamodel includes additional details.)

Attribute Coupling

System Model



MBSE Model

Interactions: Key to systems models

PIRT

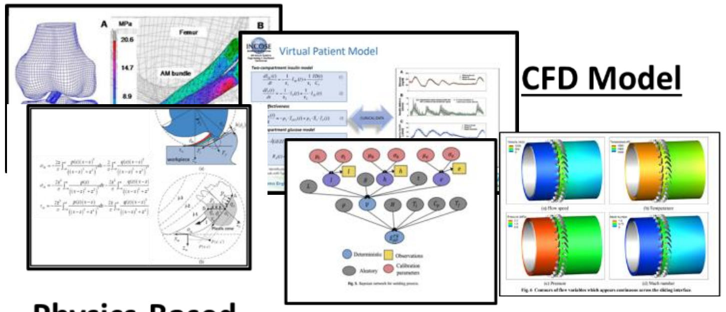
Phenomena Identification and Ranking Table (PIRT): Key to computational modeling

Computational Model

FEA Model

ODE Model

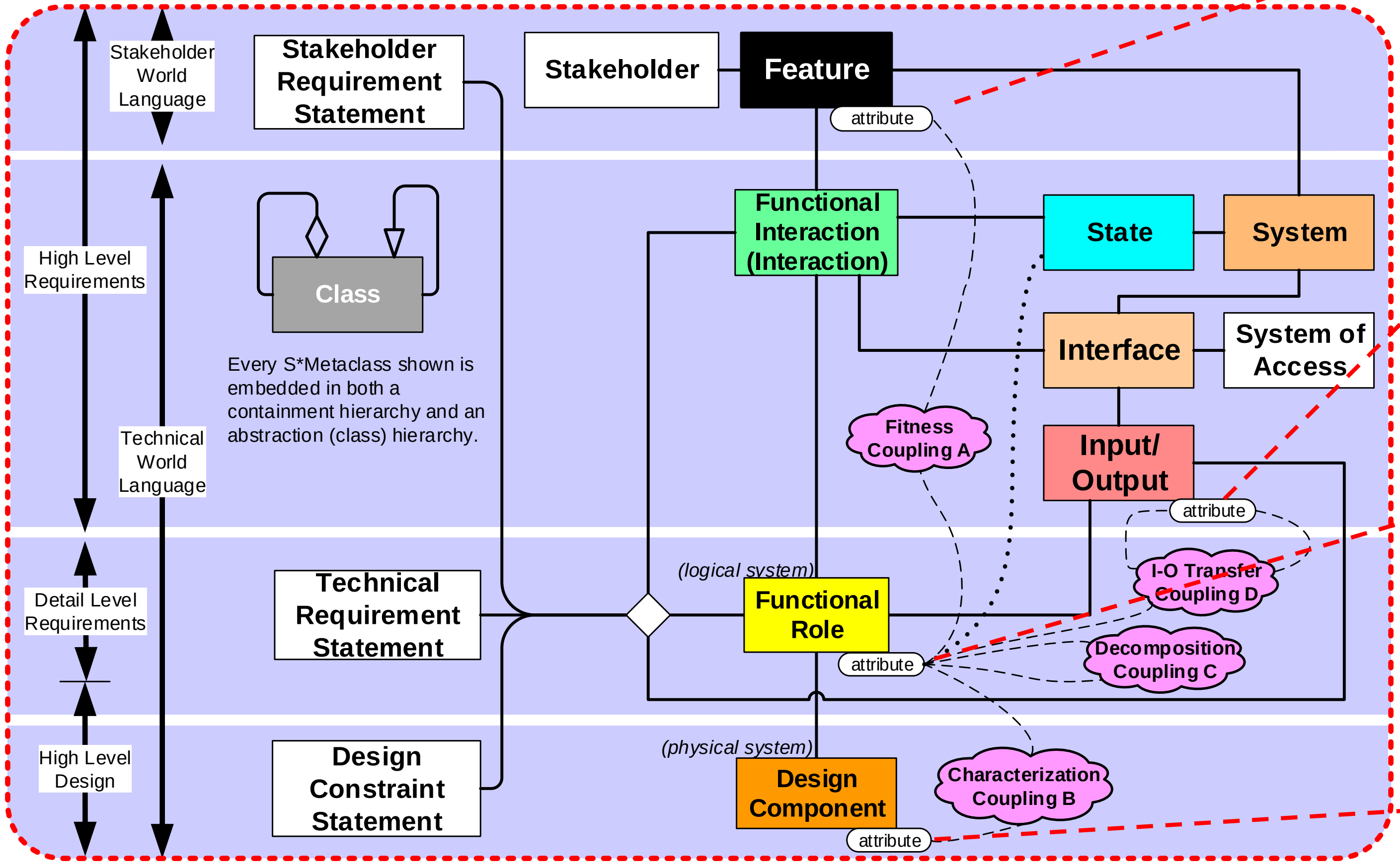
CFD Model



Physics-Based PDE Model

Data-Driven Bayesian Network Model

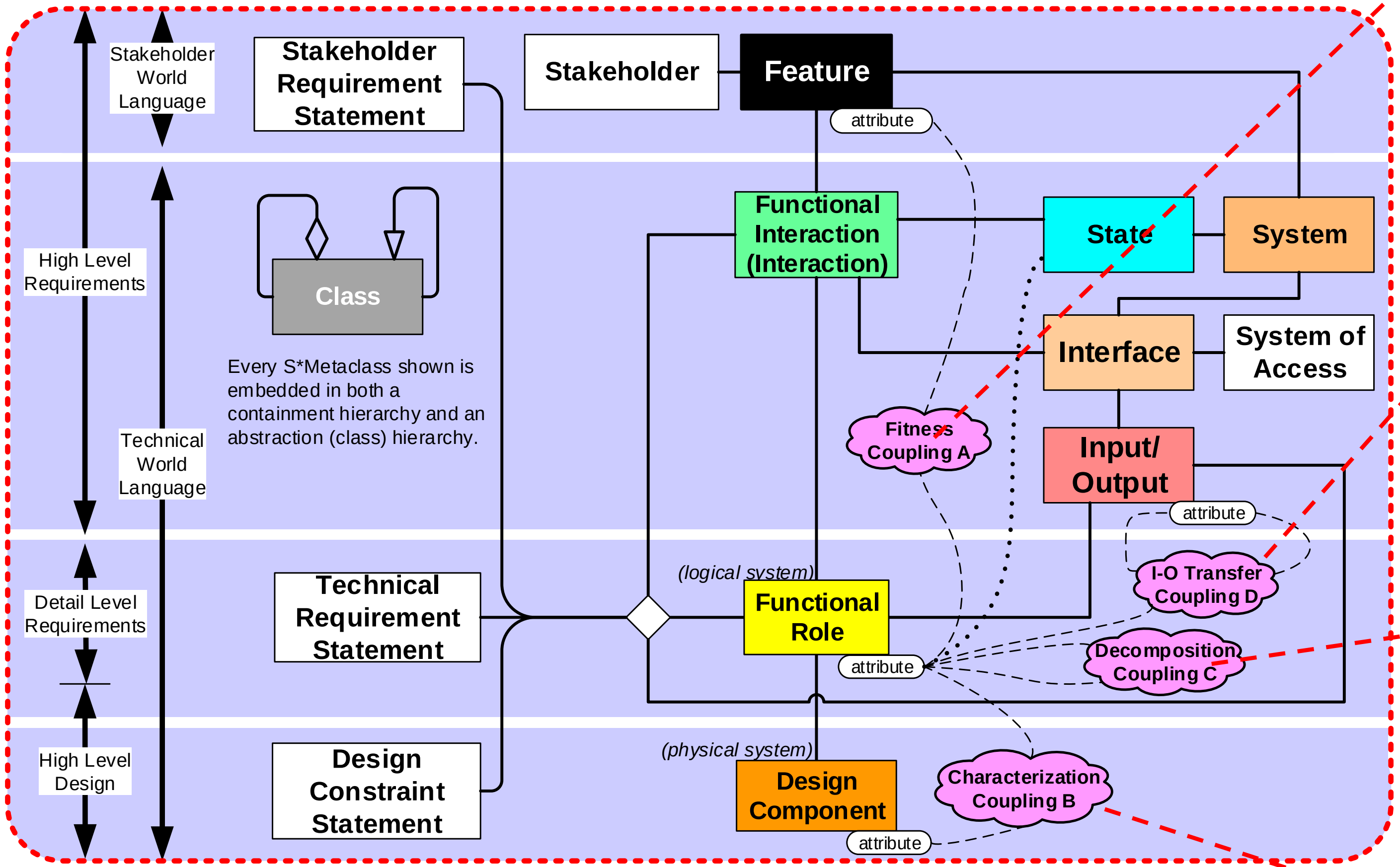
2. Attributes (variables, parameters) take on values (continuous or discrete) that quantify.



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

1. Feature Attributes quantify Measures of Effectiveness, and related stakeholder value attributes. Examples: Fuel Economy; Production Yield.
2. Input-Output Attributes quantify (often dynamical) input-output quantities. Examples: Thrust; Raw Material.
3. Role Attributes: Quantify dynamic state variables or parametric measures of performance. Examples: Tensile Strength; Melting Point; Temperature.
4. Design Component Attributes: Quantify the identity of a component to which has been allocated performance of a Functional Role. Examples: Part Number; Material Type

3. Attribute Couplings (dependencies, equations, laws) relate/constrain the values (continuous or discrete) of coupled attributes.



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

1. Fitness Couplings: Express how technical performance and stakeholder value are related—in effect, the utility or perceived value of technical performance. Examples: Market share as function of performance, cost, reliability, cost.
2. I-O Transfer Couplings: Express how output, is related to input, as function of state or other parameters. Examples: Part quality as function of raw material feedstock and process parameters.
3. Decomposition Couplings: Express how higher level system state depends on lower level subsystem parameters. Examples: Engine efficiency as function of compressor stage parameters.
4. Characterization Couplings: Express how behavior of a component is related to the identity of the component. Examples: Tensile Strength as a function of Chemical Identity.

Questions?

Contact:
info@ictt.com