Institute for Software Integrated Systems
Vanderbilt University

MODEL-INTEGRATED DESIGN IN SOFTWARE, SYSTEMS AND CONTROL ENGINEERING

Janos Sztipanovits
ISIS, Vanderbilt University

SERC Workshop

October 5, 2011

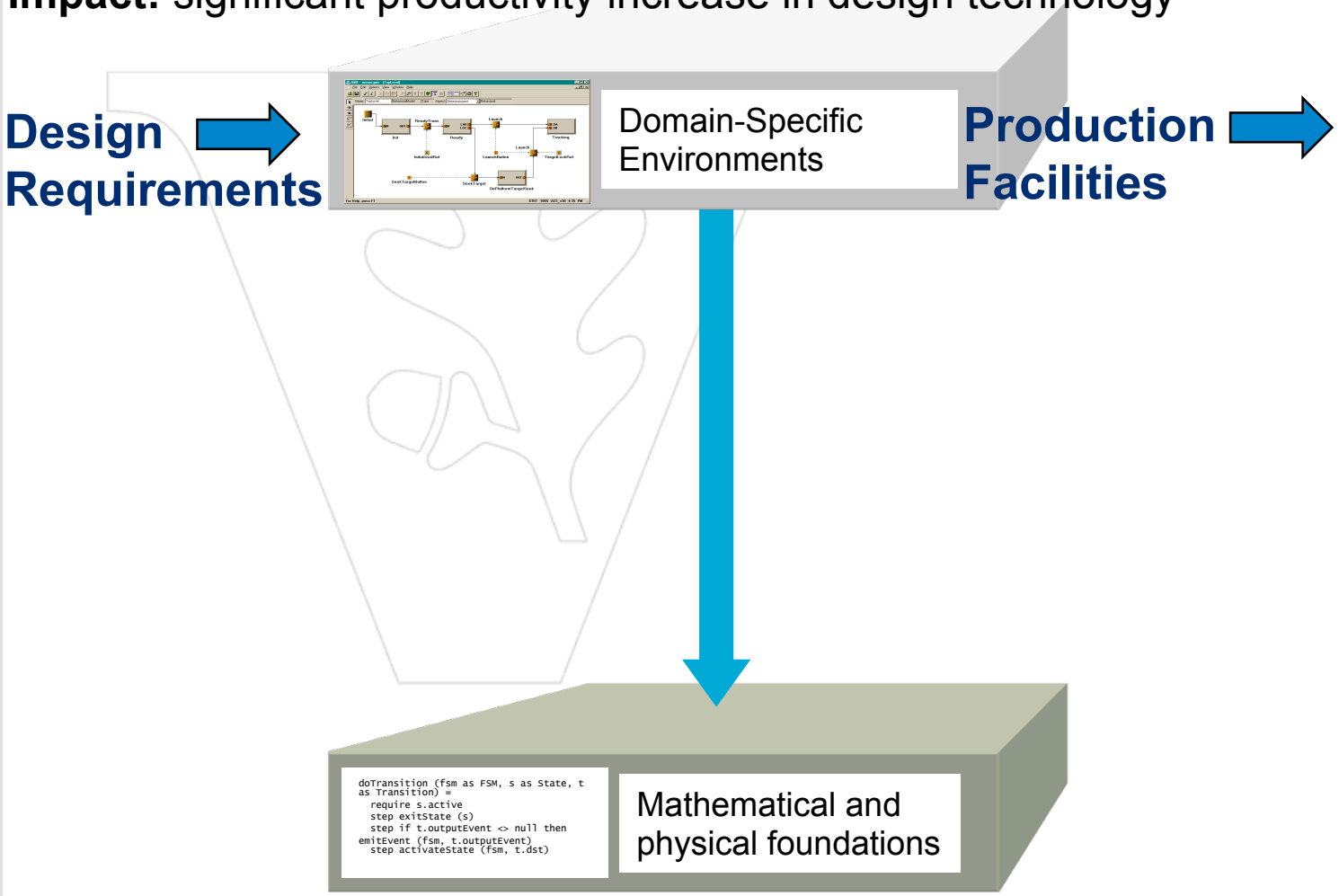
Model-Based Design Tools

Domain Specific Design Automation Environments:
- *Automotive*
- *Avionics*
- *Sensors…*

Tools:
- *Modeling*
- *Analysis*
- *Verification*
- *Synthesis*

Challenges:
- *Cost*
- *Benefit only narrow domains*
- *Island of Automation*

Key Idea: Use models in domain-specific design flows and ensure that final design models are rich enough to enable production of artifacts with sufficiently predictable properties.
Impact: significant productivity increase in design technology



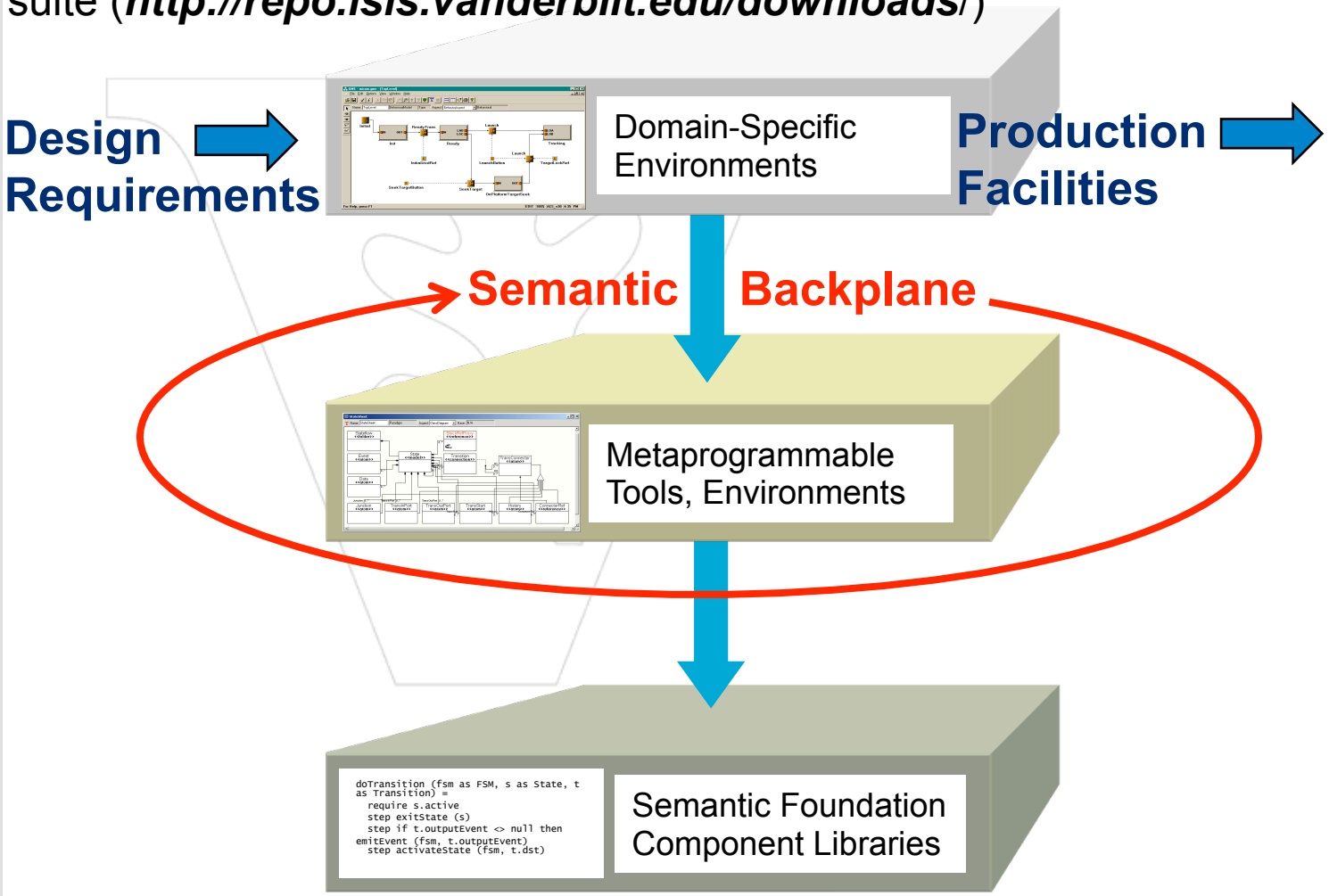
Metaprogrammable Design Tools

Domain Specific Design Automation Environments:
- *Automotive*
- *Avionics*
- *Sensors…*

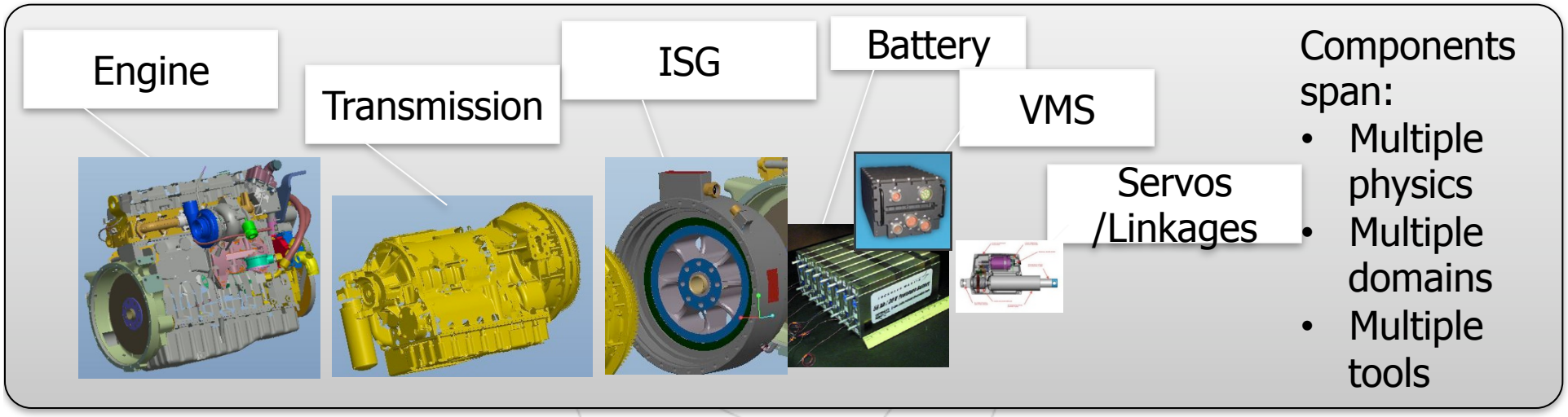
Metaprogrammable Tool Infrastructure
- *Model Building*
- *Model Transf.*
- *Model Mgmt.*
- *Tool Integration*

Explicit Semantic Foundation
- *Structural*
- *Behavioral*

Key Idea: Ensure reuse of high-value tools in domain-specific design flows by introducing a metaprogrammable tool infrastructure.
VU-ISIS implementation: Model Integrated Computing (MIC) tool suite (***http://repo.isis.vanderbilt.edu/downloads***/)

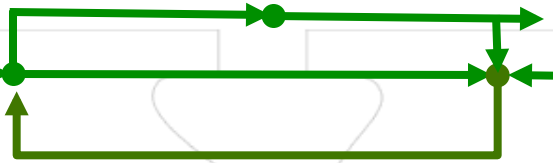
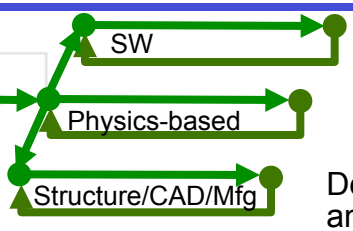


Use Case 1: Cyber Physical Systems



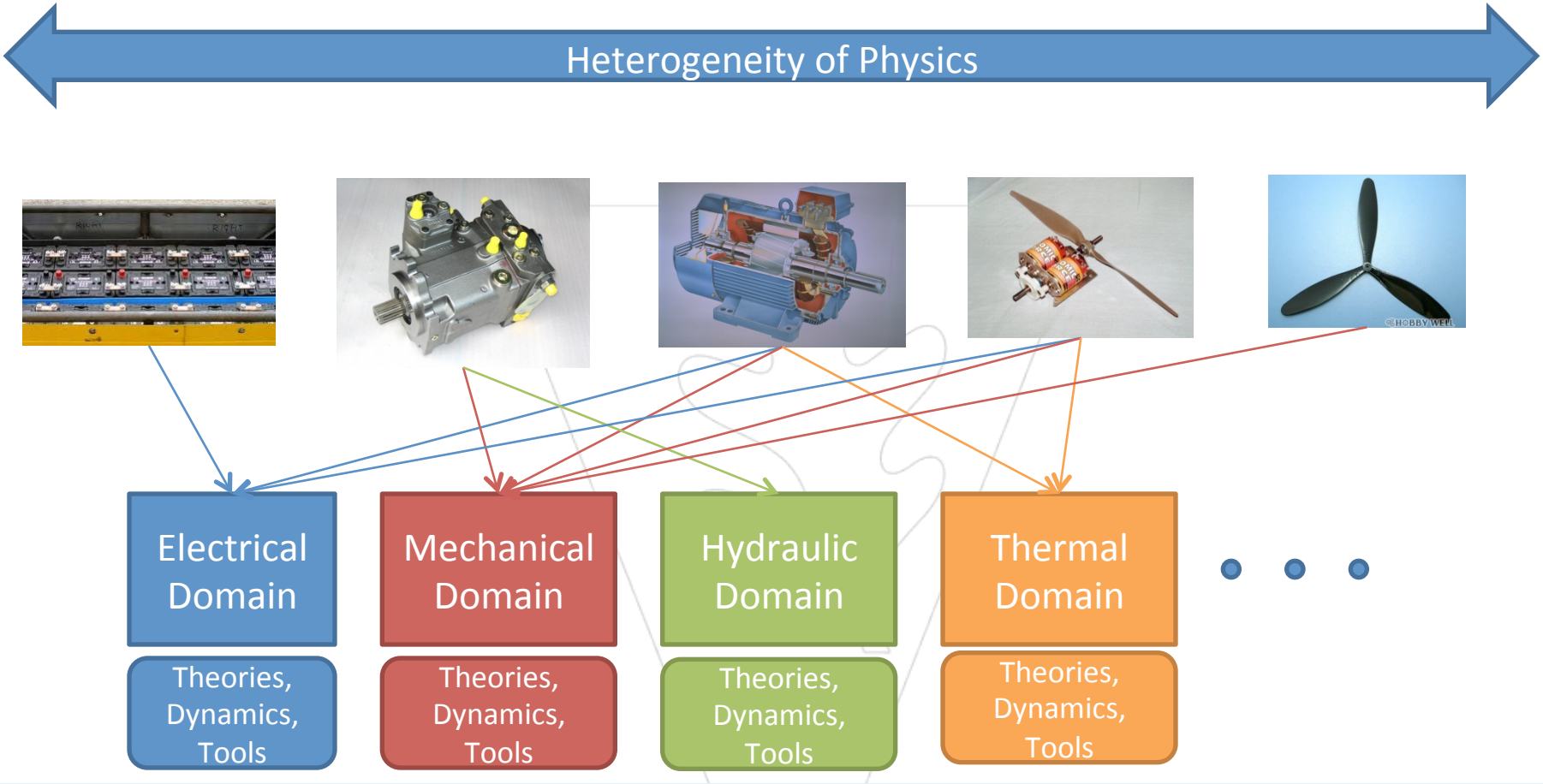
- Physical
  - Functional: implements some function in the design
  - Interconnect: acts as the facilitators for physical interactions
- Cyber
  - Computation and communication that implements some function
  - Requires a physical platform to run/to communicate
- Cyber-Physical
  - Physical with deeply embedded computing and communication

CPS Design Flow Requires Model Integration

Architecture Design		Integrated Multi-physics/Cyber Design			Detailed Design	
Modeling	**Exploration**	**Modeling**	**Simulation**	**V&V**	**Modeling**	**Analysis**
						
Rapid exploration		Exploration with integrated optimization and V&V				
• Architecture Modeling • Design Space + Constraint Modeling • Low-Res Component Modeling		• Architecture Modeling • Design Space + Constraint Modeling • Dynamics Modeling (ODE) • Computational Behavior Modeling • CAD/Thermal Modeling • Manufacturing Modeling			• Architecture Modeling • Dynamics, RT Software, CAD, Thermal, … • Detailed Domain Modeling (FEM)	

Domain Specific Modeling Languages

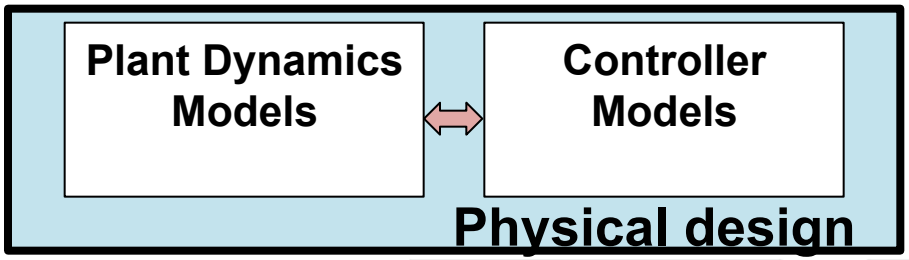
Model Integration Challenge: Physics



Physical components are involved in multiple physical interactions (multi-physics)

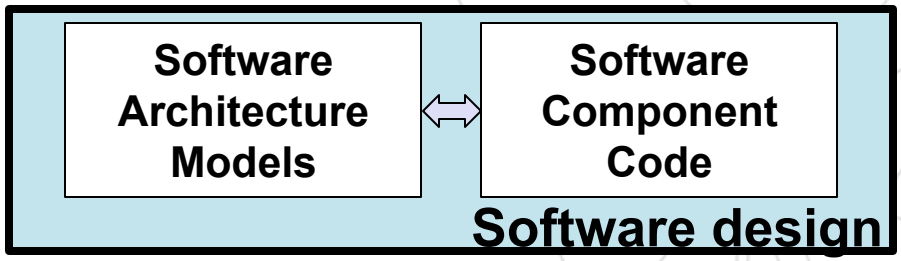
Source of resilience: **explicit modeling of multi-physics interactions.**

Model Integration Challenge: Implementation Layers



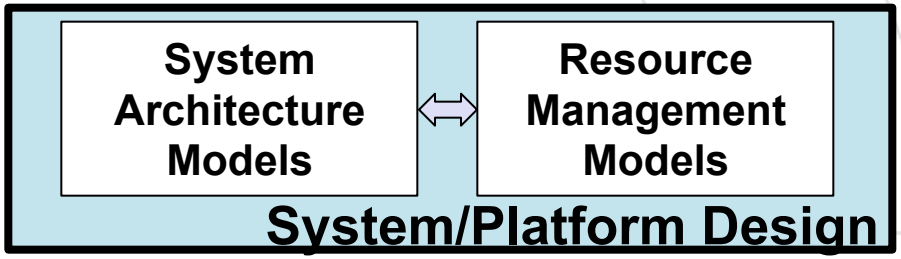
Dynamics: $B(t) = \kappa_p(B_1(t),\ldots,B_j(t))$

- *Properties*: stability, safety, performance
- *Abstractions*: continuous time, functions, signals, flows,…



Software : $B(i) = \kappa_c(B_1(i),\ldots,B_k(i))$

- *Properties*: deadlock, invariants, security,…
- *Abstractions*: logical-time, concurrency, atomicity, ideal communication,..

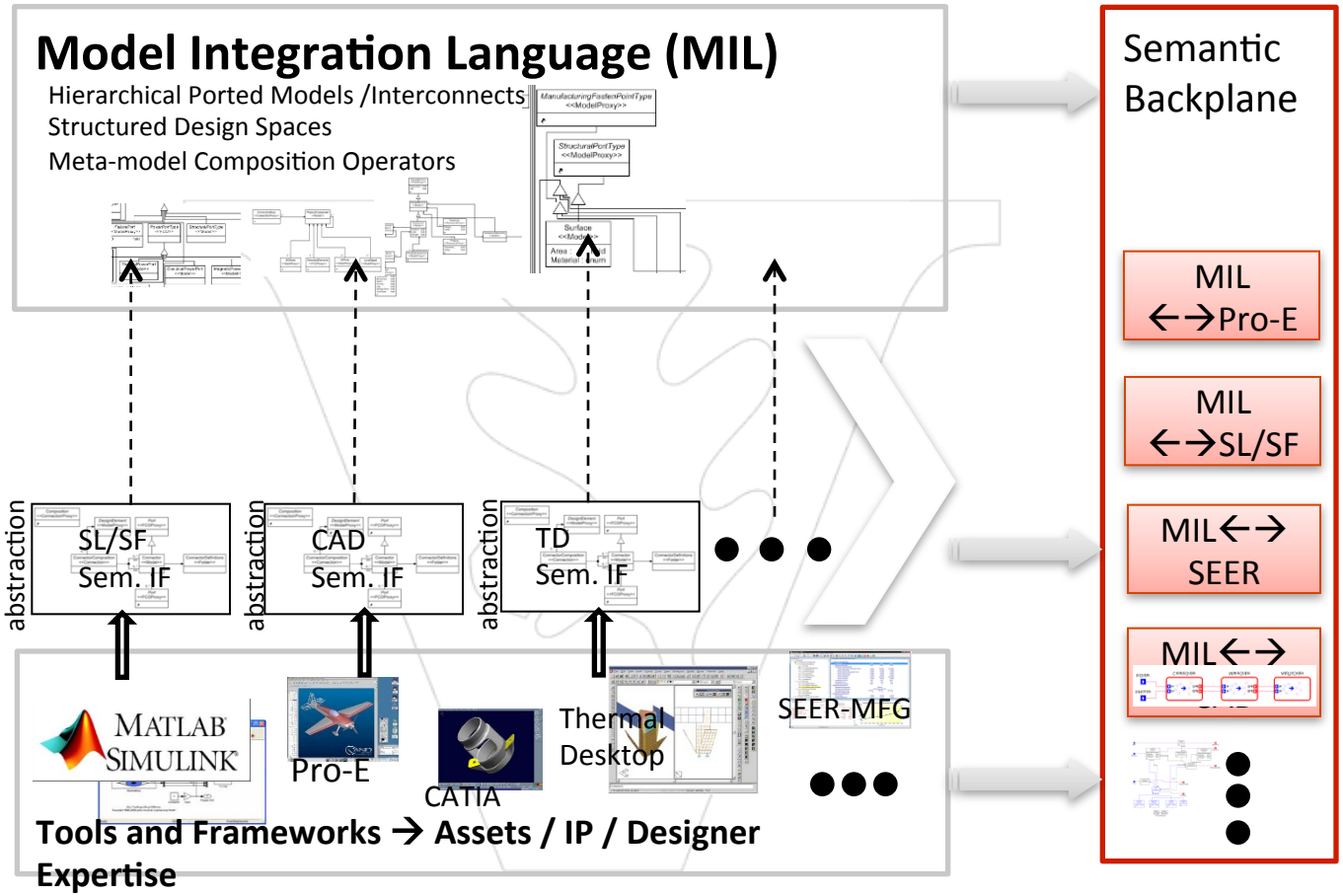


Systems : $B(t_j) = \kappa_p(B_1(t_i),\ldots,B_k(t_i))$

- *Properties*: timing, power, security, fault tolerance
- *Abstractions*: discrete-time, delays, resources, scheduling,

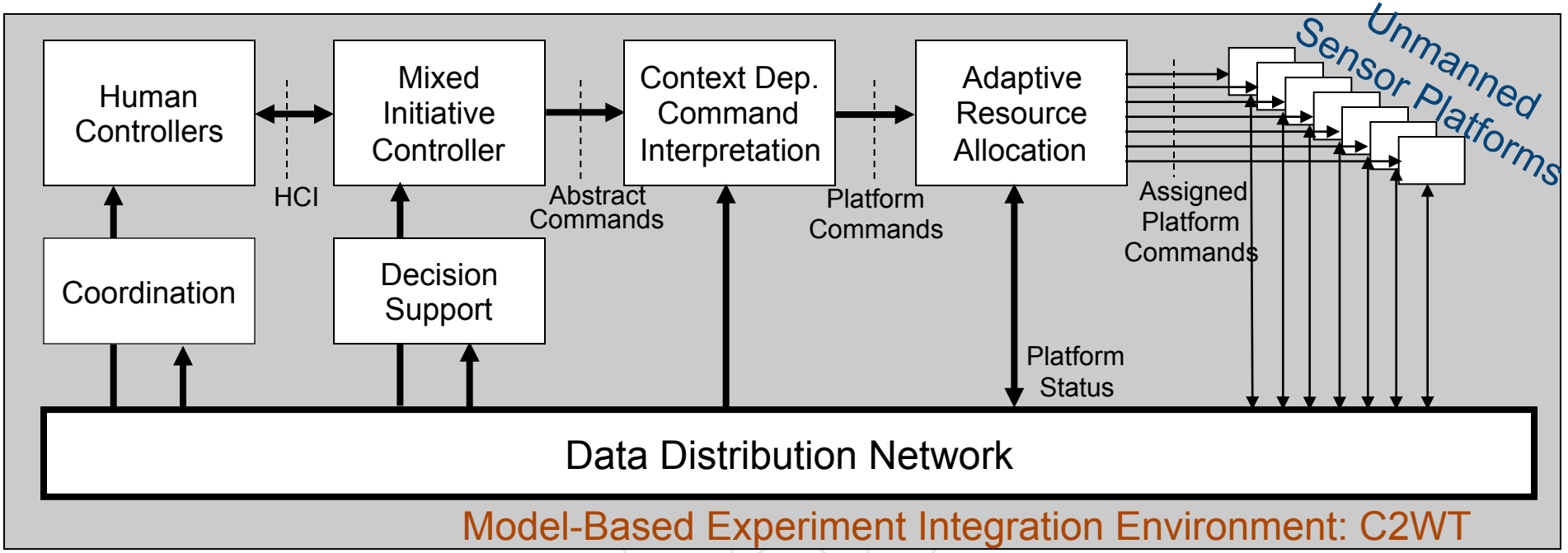
Source of resilience: **systems science principles for decoupling across design layers (such as passive dynamics to decouple stability from implementation induced time-varying delays**

Model Integration Language



Impact: Open Language Engineering Environment → Adaptability of Process/Design Flow → Accommodate New Tools/Frameworks , Accommodate New Languages

Use Case 2: "C2 Wind Tunnel"



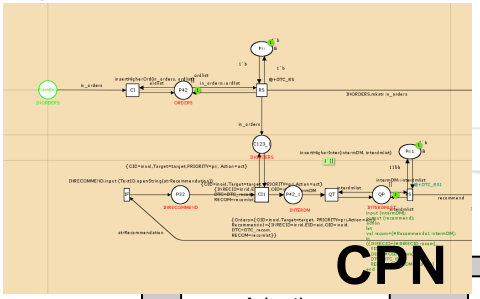
Issues to be studied experimentally:

- **Distributed Command and Control**
  - Synchronization and coordination
  - Distributed dynamic decision making
  - Network effects
- **Information Sharing**
  - Shared situation awareness
  - Common Operation Picture (COP)
  - Network effects
- **Advanced Cooperative Control**
  - Cooperative search algorithms

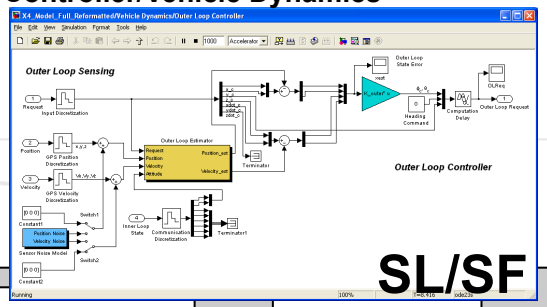
AFOSR PRET Program

Heterogeneous Simulation Integration

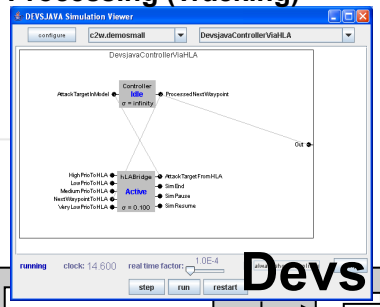
Organization/Coordination



Controller/Vehicle Dynamics



Processing (Tracking)



3-D Environment (Sensors)

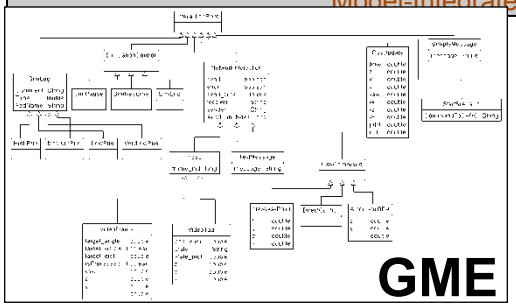
Adaptive Human | Mixed Initiative | Context Dep. Command | Adaptive Resource

How can we integrate the models?
How can we integrate the simulated heterogeneous system components?
How can we integrate the simulation engines?

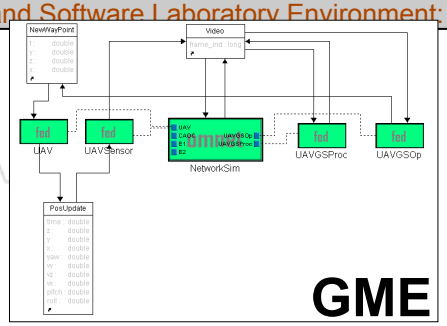
Elements | Elements | Elements | Status

Data Distribution Network

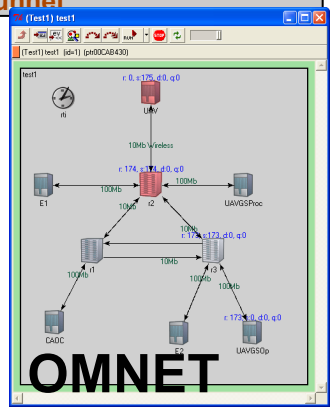
Model-Integrated System and Software Laboratory Environment: **C2 Windtunnel**



Simulation Interaction

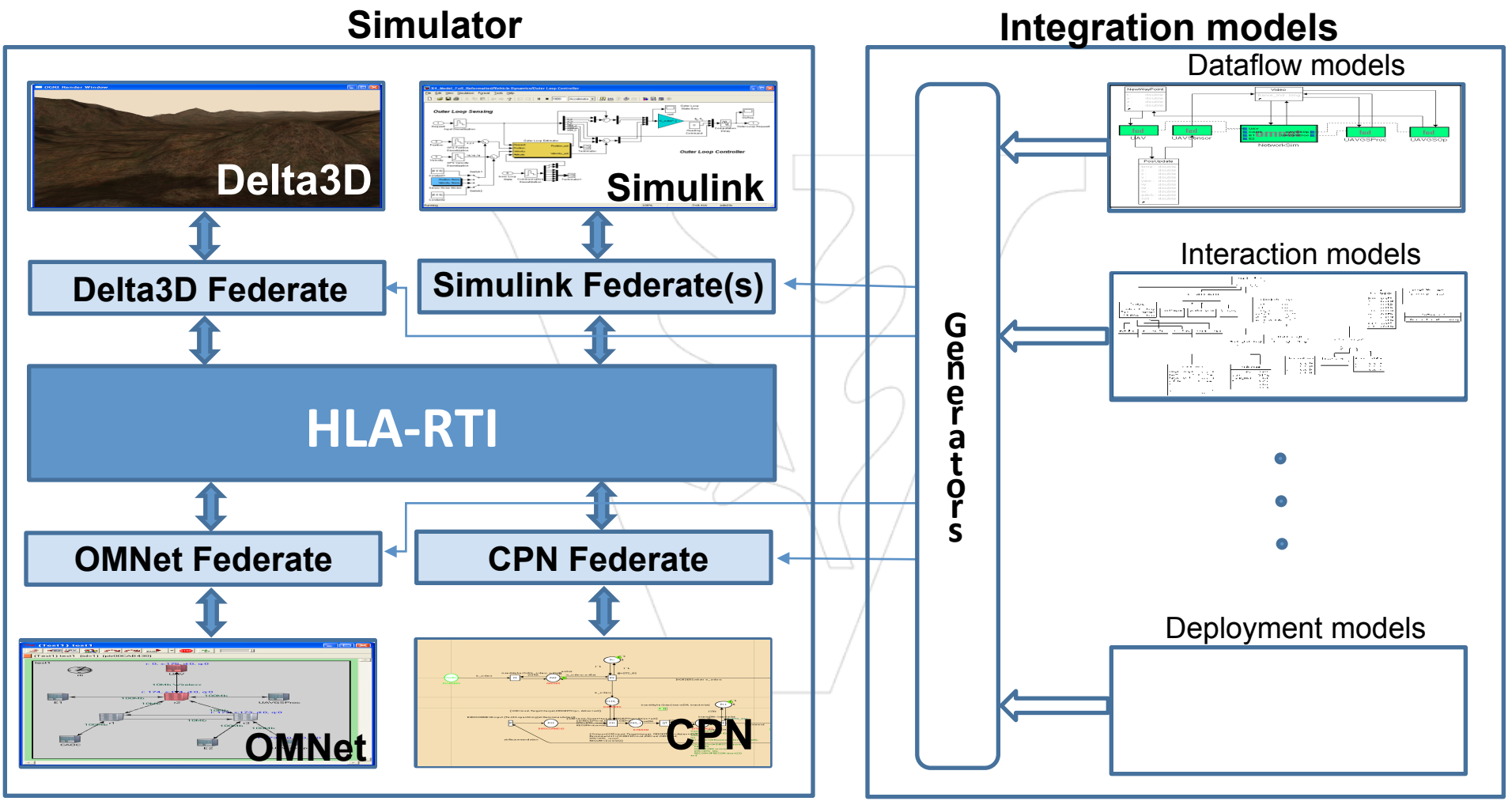


Simulation Architecture

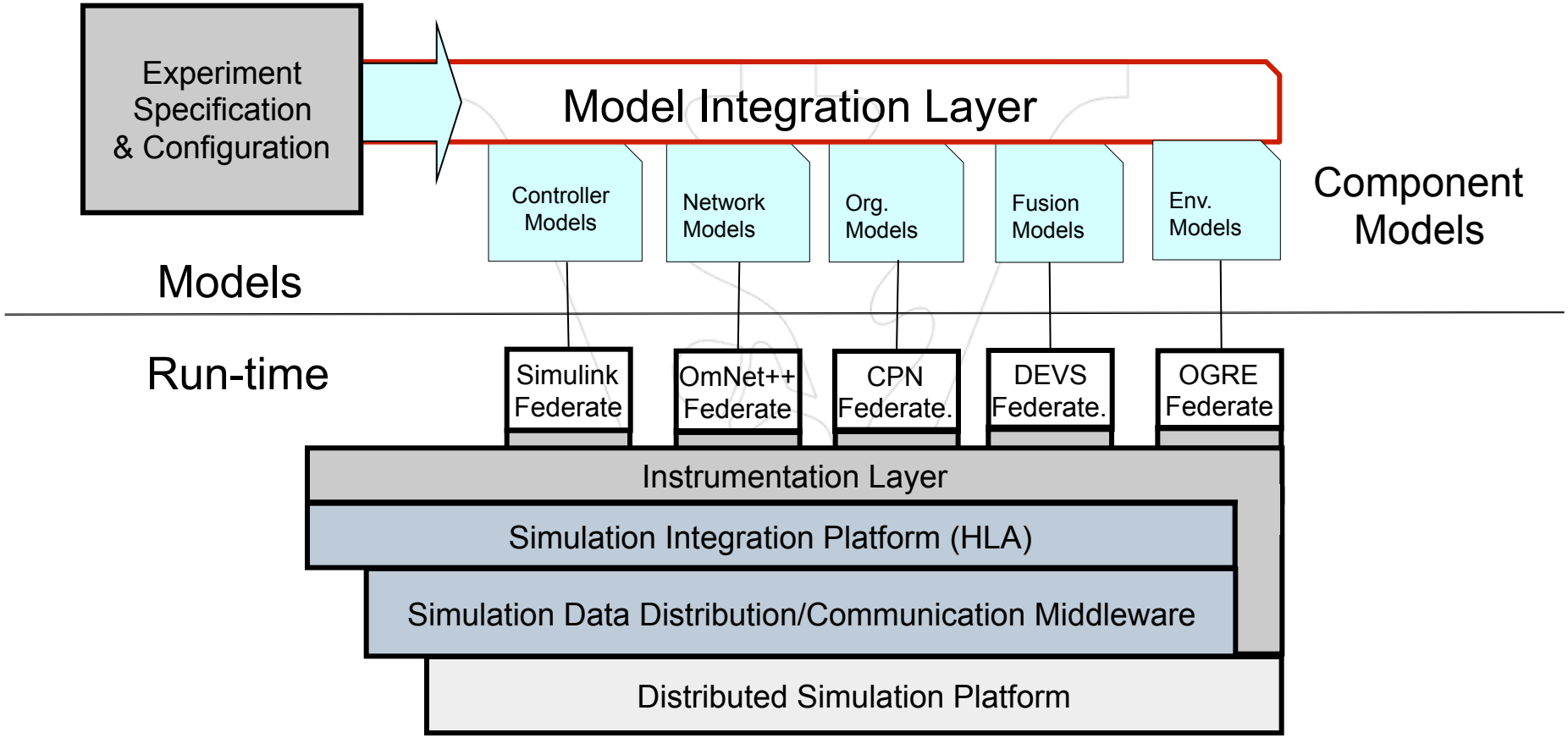


Network Architecture

Model Integration Architecture in C2WT

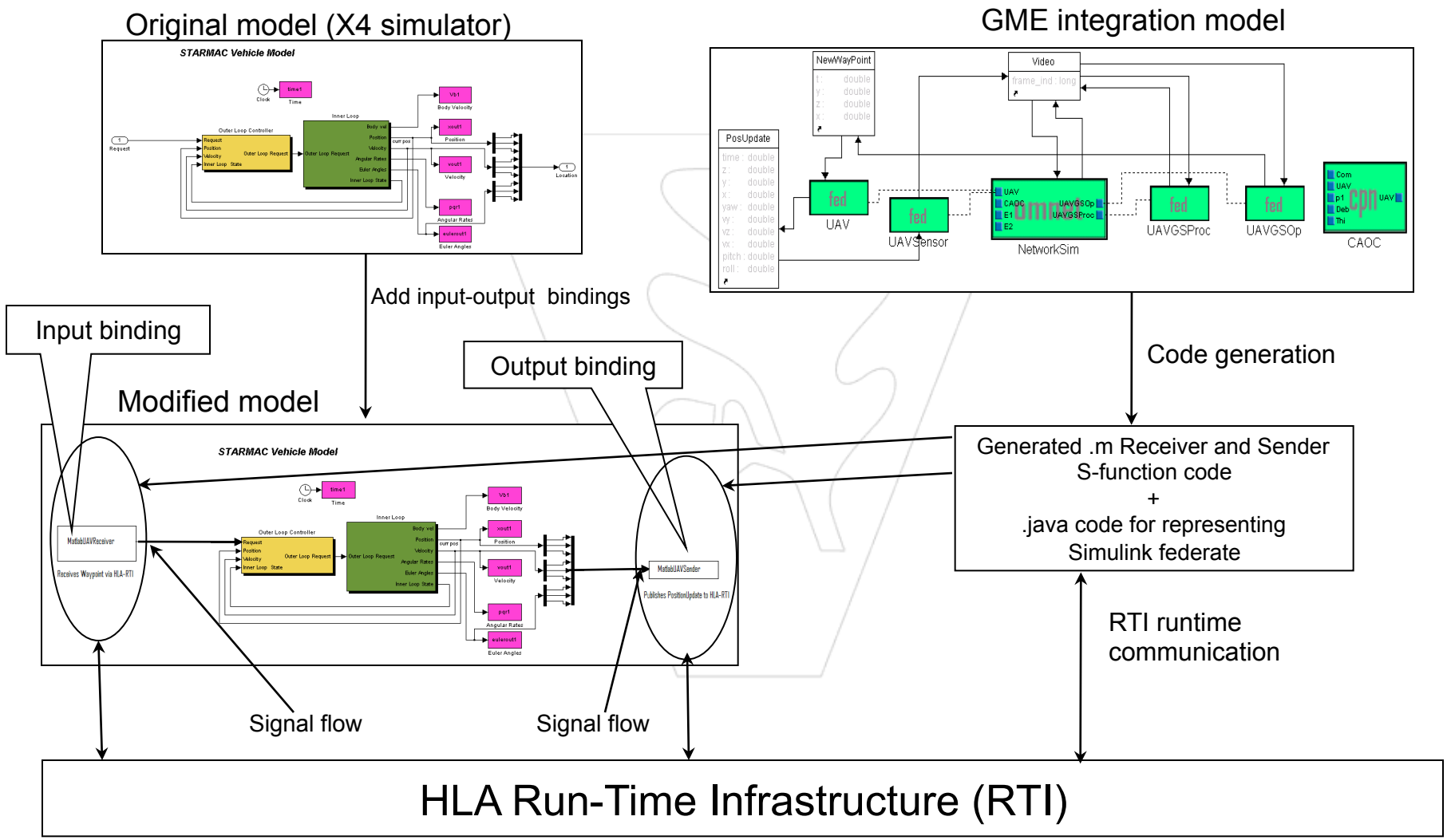


Simulation Integration Architecture in C2WT



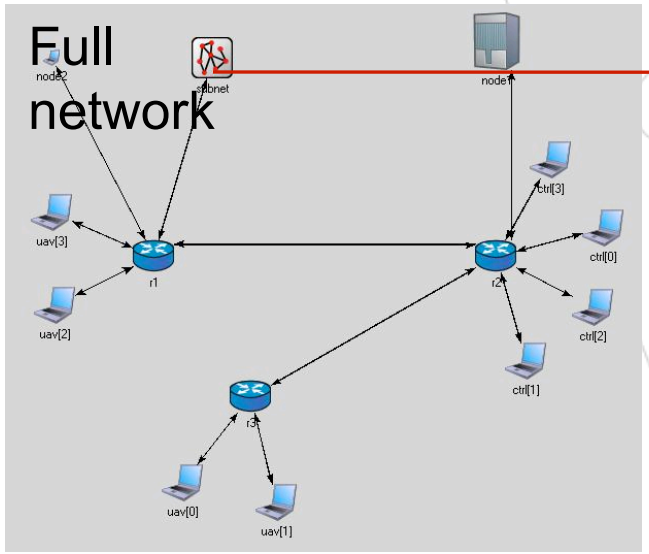
https://wiki.isis.vanderbilt.edu/OpenC2WT

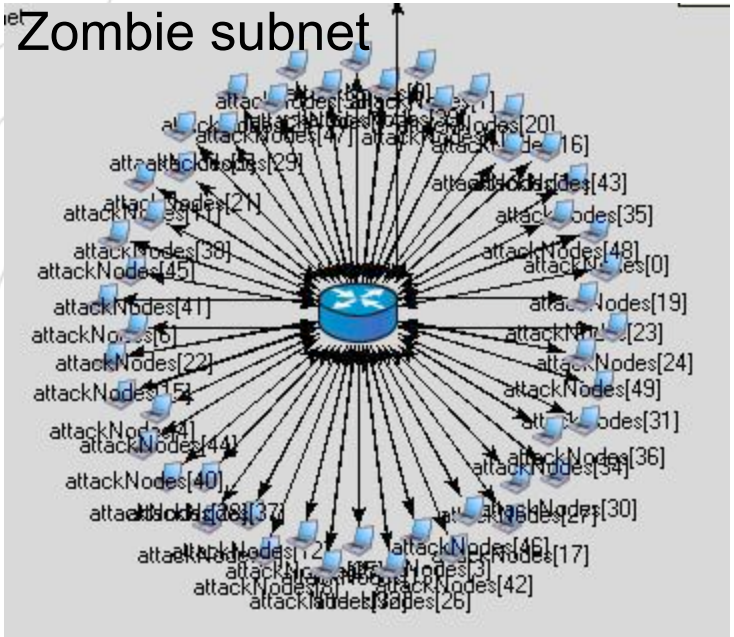
Example: Simulink model integration (Vehicle dynamics)



Experiments: Impact of Cyber Attacks

- **Network attack:**
  - A sub-network with hundreds of zombie nodes attacks a critical router on the main network.
  - Flood attack on udp, tcp or ping



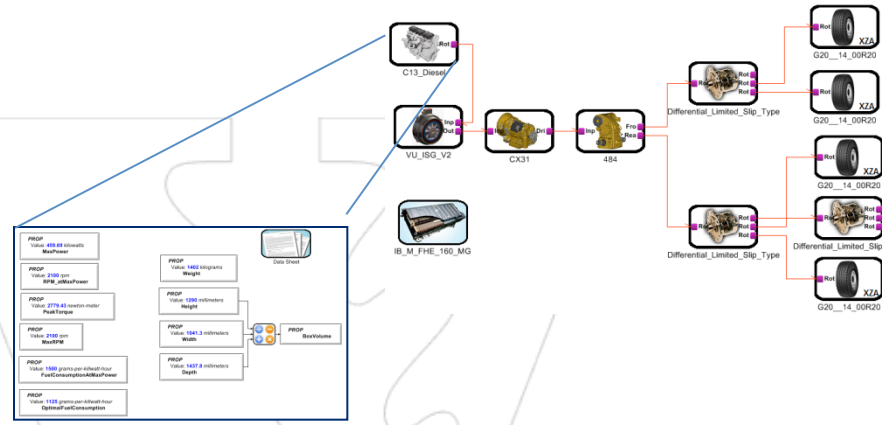


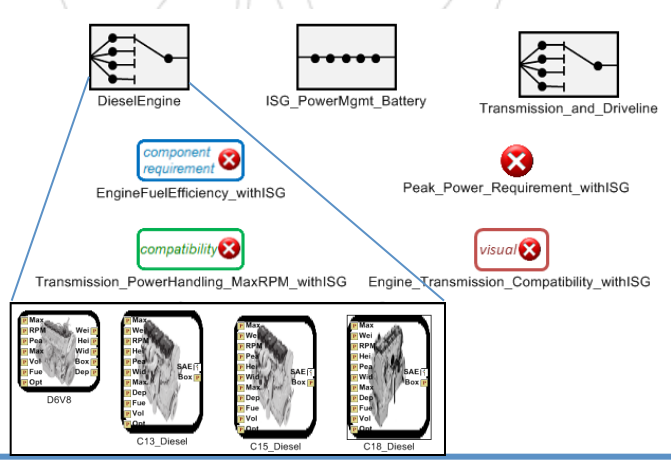
Summary

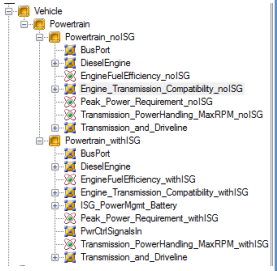
- Questions:
  - What are challenging systems application domains?
    Heterogeneous SoS domains (like CPS and C2).
  - How does practice diverge from theory, and how do we connect?
    Precise compositionality is hard to achieve in heterogeneous systems, still, we need predictability. Need systems science principles for simplifying interactions and dependences (decoupling).
  - Where are relevant technologies to be found?
    In cross-disciplinary interactions. E.g. scalability in embedded software verification may require tradeoffs in systems dynamics.
  - What would be the most critical tools and products?
    Component-based and model-based design approaches and tools are and will be increasingly essential.

Example: Architecture Modeling

Sublanguage / Capability	Formalism, Language Constructs, Examples	Usage
Architecture Modeling	Hierarchical Module Interconnect - Components - Interfaces - Interconnects - Parameters - Properties	Systems Architect - Explore Design Space - Derive Candidate Designs
Design Space Modeling	Hierarchically Layered Parametric Alternatives - Alternatives/ Options - Parameters - Constraints	Systems Architect - Define Design Space - Define Constraint

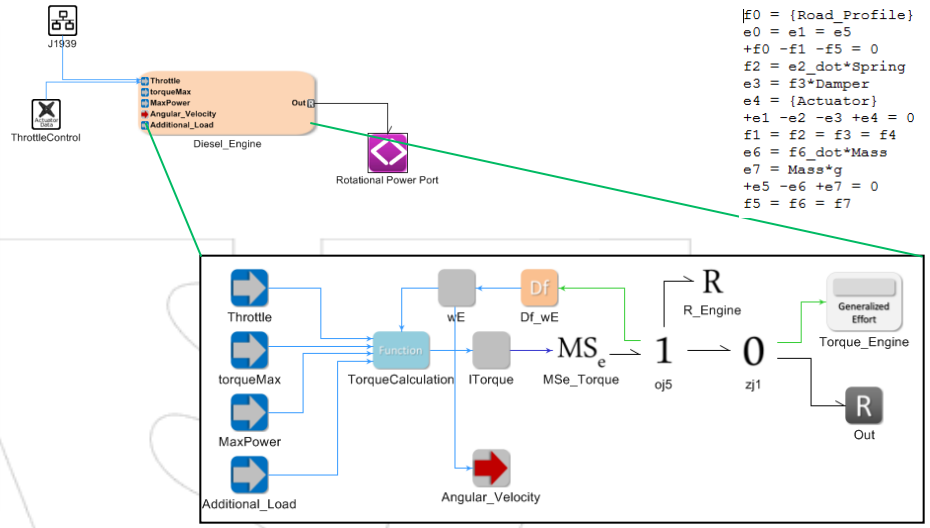


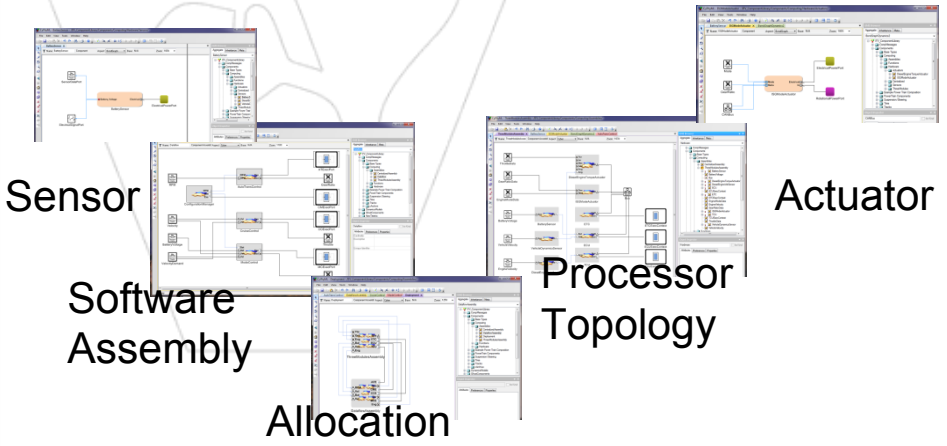




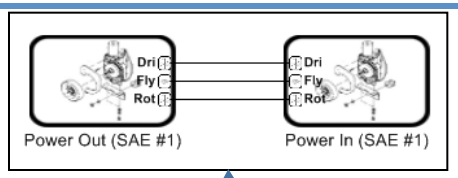
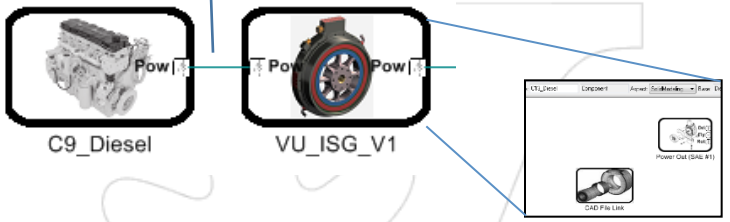
Example: Dynamics Modeling

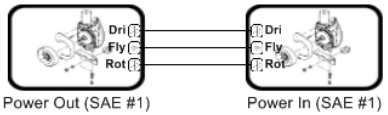
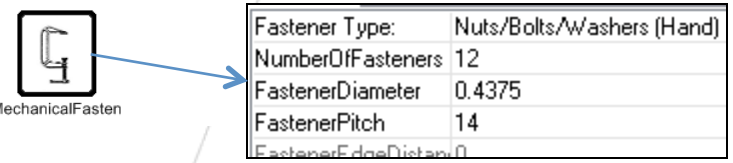
Physical Dynamics Modeling	Hybrid Bond Graphs - Efforts, Flows, - Sources, Capacitance, Inductance, - Resistance, - Transformers Gyrators,	Component Engineer - model dynamics with Hybrid Bond Graphs System Engineers - Compose system dynamics
Computational Dynamics Modeling	Dataflow + Stateflow + TT Schedule - Interaction with Physical Components - Cyber Components - Processing Components	Domain Engineers - design controller System Engineers - Processor allocate - Platform Effects





Example: Physical Structure and Manufacturing Modeling

Solid Modeling (CAD / Geometry)	Structural Interfaces
	- Defined with Peer Roles:
	  - Axis
	  - Point
	  - Surface
	- CAD Links
		
		Standard Structural Interfaces (ex: SAE #1)
		 
			Component Engineer
			-Defines Structural Interface
			System Engineer
			- Defines Architecture

Manufacturing Modeling	Component Manuf. Cost
	-Make
	  - Material
	  - Fab Proc
	  - Complxity
	  - Shape/Wt
	-OTS: Cost/unit
	Structural Interfaces
	-Fastener Types, Num# …
		
		 
			Component Engineer
			-Defines Part Cost
			-Defines Structural Interface, Fastener