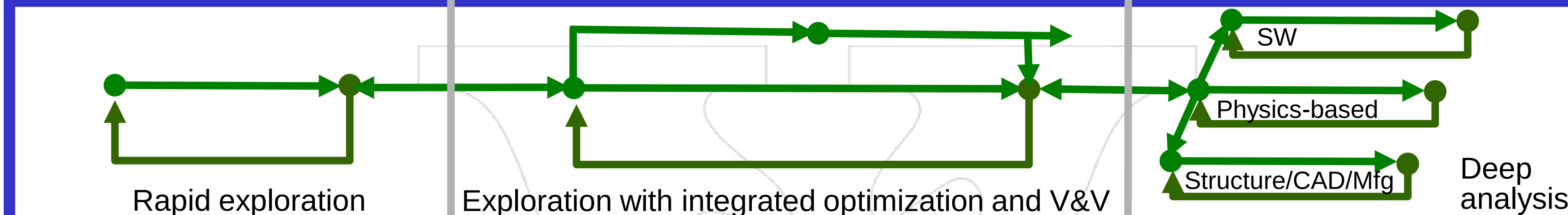
# Lessons Learned in Model-Based Design: Semantics is Important

**Janos Sztipanovits and**
**Tihamer Levendovszky**

Institute for Software Integrated Systems
Vanderbilt University
Email: janos.sztipanovits@vanderbilt.edu

# Design Flow Requires Model Interchange

| Architecture Design | | Integrated Multi-physics/Cyber Design | | | Detailed Design | |
|---|---|---|---|---|---|---|
| Modeling | Exploration | Modeling | Simulation | V&V | Modeling | Analysis |




**Architecture Design**

- Design Space + Constraint Modeling
- Architecture Modeling
- Static Component Modeling (multiphysics)

**Integrated Multi-physics/Cyber Design**

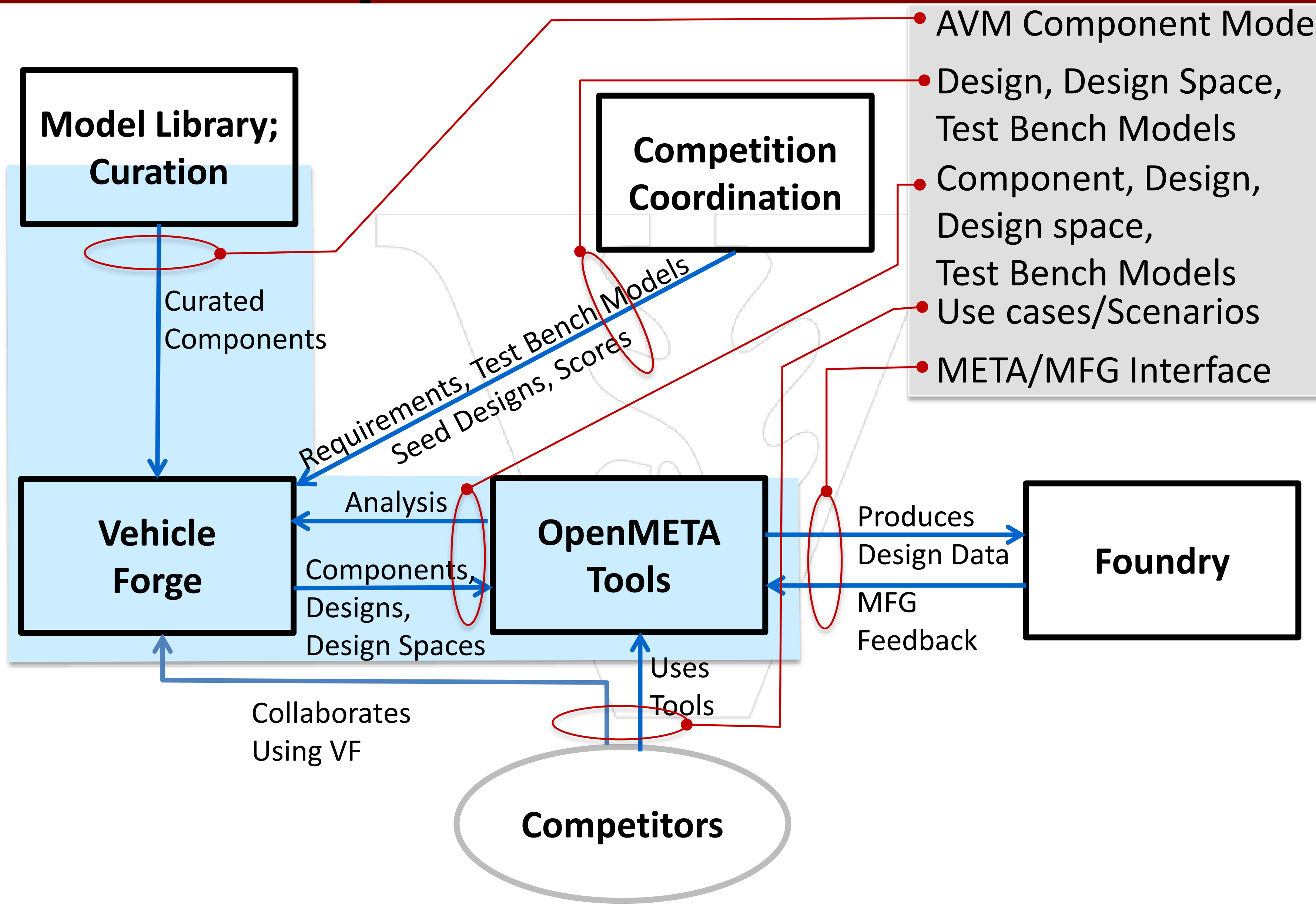
- Design Space + Behavioral Constraint Modeling
- Architecture Modeling
- Dynamics Modeling (multiple abstractions and multiphysics)
- CAD/assembly modeling
- Coarse Manufacturing Constraint Modeling

**Detailed Design**

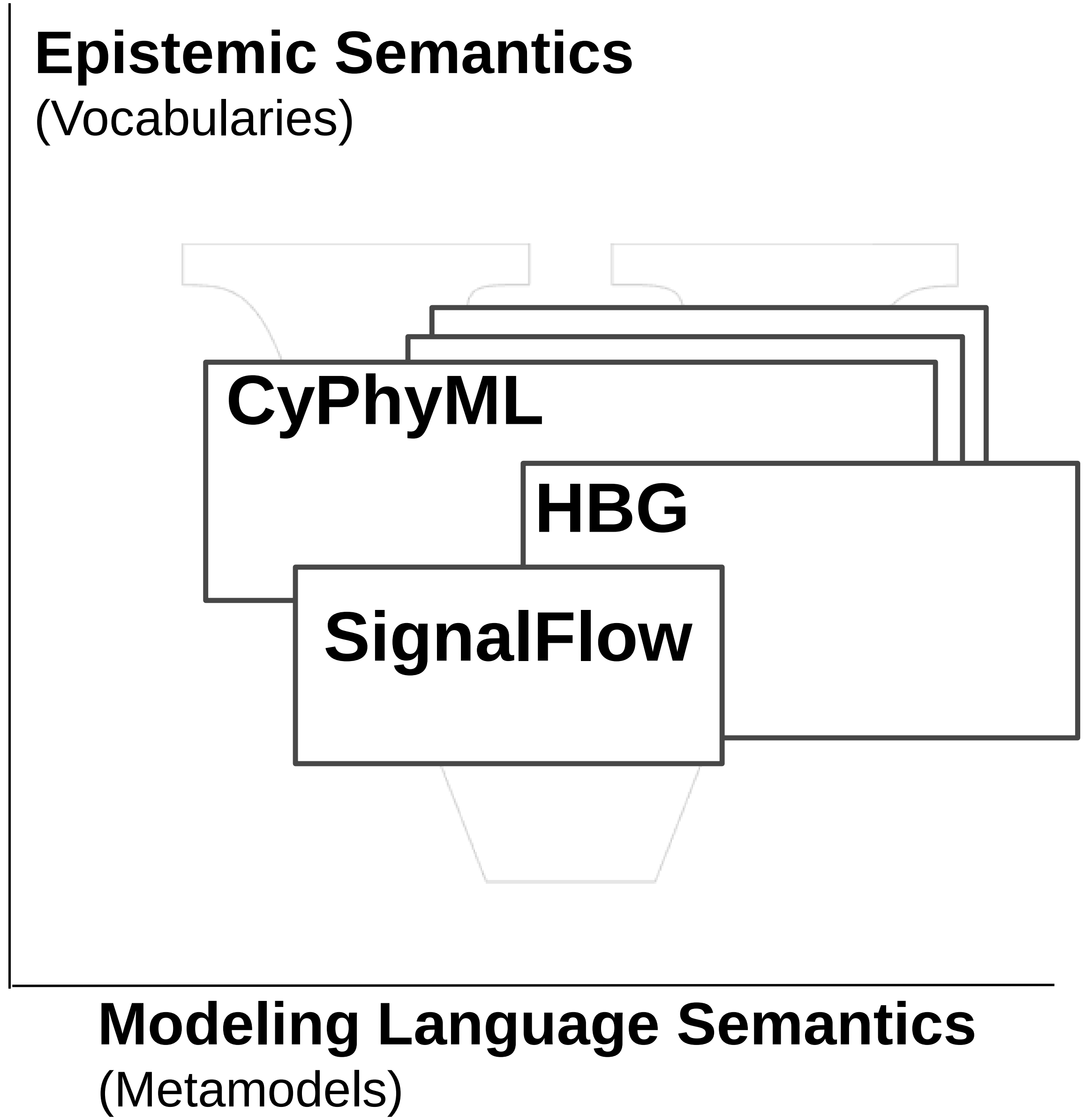
- Architecture Modeling
- Detailed Domain Modeling
  - CAD
  - FEA; thermal, fluid…
  - Surrogate gen.
- Detailed Mfg. modeling
- RT SW modeling

Domain Specific Modeling Languages

# Example: AVM Systems Require Complex Information Flows



# OpenMETA Information Architecture



# Information Architecture

## Models and Modeling Languages

### Design Data Package (DDP)

- Component Model
- Design Model
- Design Space Model
- Requirement Model
- Result Package

- CyPhy Model Integration Language
- Test Bench Integration Language

- Embedded System Modeling Language (ESMOL)
- Modelica
- DESERT
- CAD
- FEA
- Parametric Exploration Tool (PET)

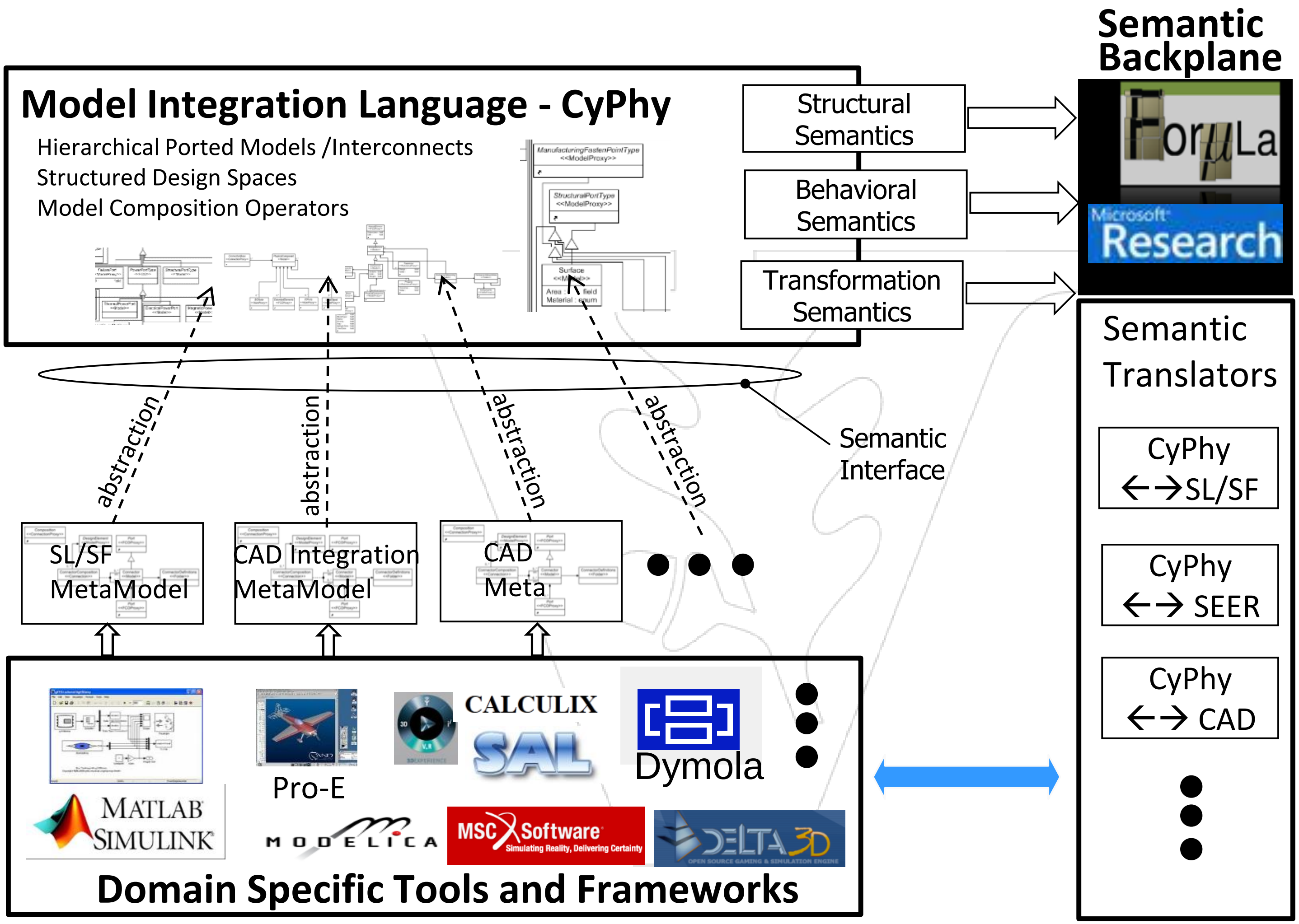
- Signal Flow Modeling Language
- Software Architecture Modeling Language
- Deployment Modeling Language
- Software Component Modeling Language
- Bond Graph
- Qualitative Abstraction
- Relational Abstraction
- Probab. Analysis (PCC)
- Fault Modeling

## Standardized Vocabularies and Core Types

- META Ontologies
- VehicleForge Ontology
- iFAB Ontology

- Interface & Composition Vocabulary
- Behavior Vocabulary
- Testing Vocabulary
- Vehicle Component Vocabularies
- <<note>> In progress. Currently includes characterizations of supplier data (unknown source for this vocab)

# Case for Model Integration Languages…




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

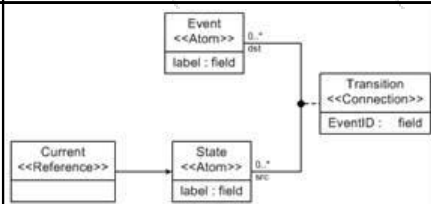
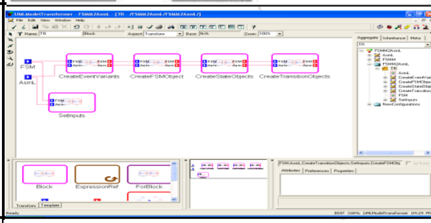
# Convergence in Formal Framework: FORMULA

- History: Foundations for Embedded Systems ITR; Ethan Jackson at VU 2005-2008
- Microsoft Research (Bellevue & Aachen); Satisfiability Modulo Theory Solver (Z3); VS distribution
- *http://research.microsoft.com/formula*

- Foundation: Algebraic Data Types (ADT) and First-order logic with fixpoints (FPL)
- Parameterized with background theories (bit vectors, term algebras, etc.
- Semantics is defined by constraint logic programming (CLP)
- Evolving structures; temporal logic

# Current Work: Semantic Backplane

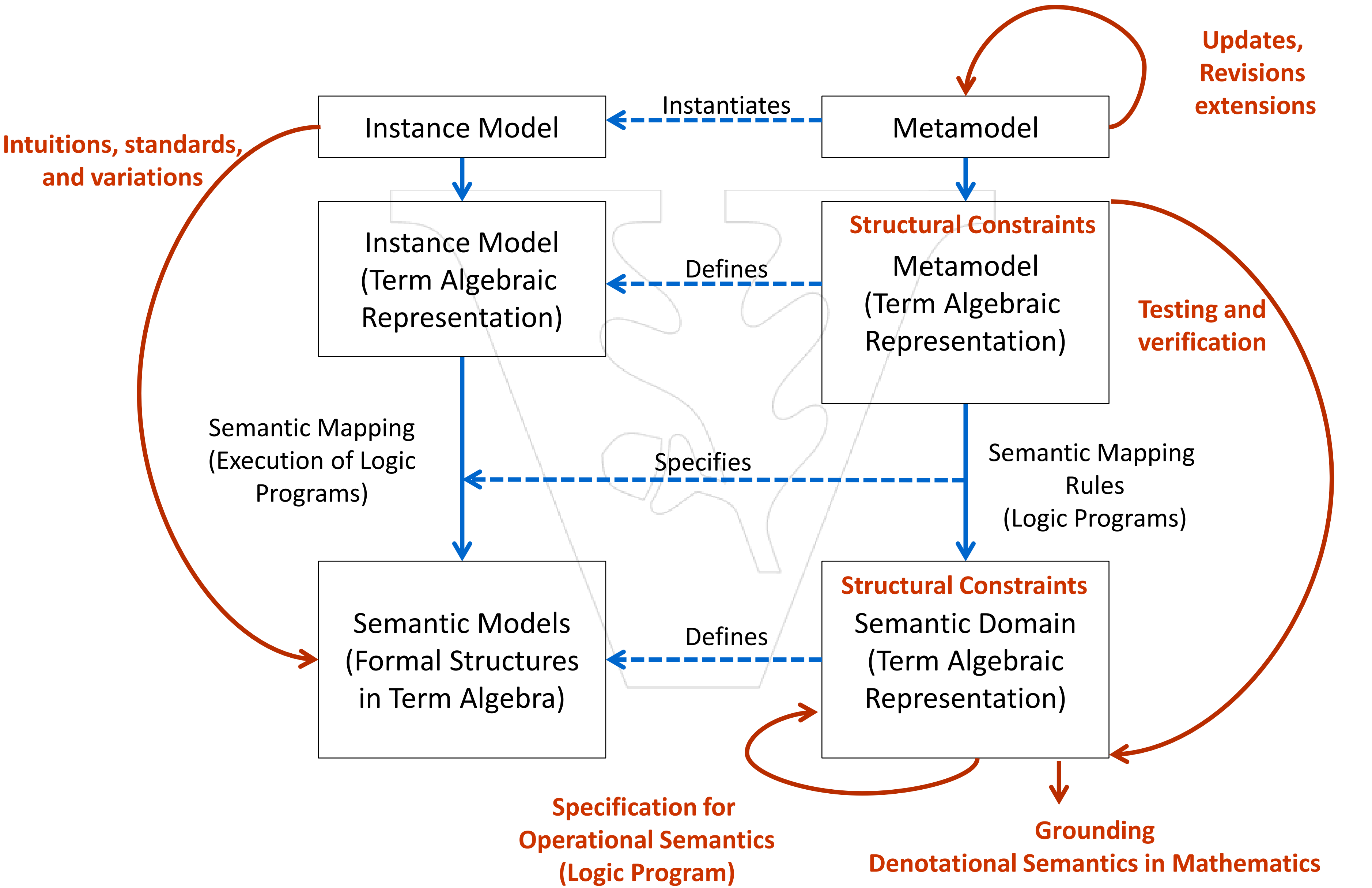
The Semantic Backplane is based on a mathematical framework provided by term algebra and logics, incorporates a tool suite for specifying, validating and using formal structural and behavioral semantics of modeling languages, and includes a library of metamodels and specifications of model transformations.

<table>
<tr><th>Functions</th><th>(Meta)Models</th><th>Languages</th><th>Tools</th><th>Role</th></tr>
<tr><td>Metamodeling</td><td>
</td><td>MetaGME</td><td>• GME<br>• MetaGME-2-Formula</td><td>• DSML spec.<br>• Constraint Checking<br>• Metaprog.</td></tr>
<tr><td>Transformation Modeling</td><td></td><td>UMTL</td><td>• GReAT<br>• UDM</td><td>• Transf. spec.<br>• Compiling spec to transformer</td></tr>
<tr><td>Formal Metamodeling</td><td><pre>domain DFA {
  primitive Event ::= (lbl: Integer).
  primitive State ::= (lbl: Integer).
  [Closed(src, trg, dst)]
  primitive Transition ::= (src: State,
  [Closed(st)]
  primitive Current     ::= (st: State).</pre></td><td rowspan="2">Formula (MSR)</td><td>• Domain Comp.<br>• Trace Gen.</td><td>• Metamod. checking<br>• Example gen.<br>• Semantic units</td></tr>
<tr><td>Formal Transformation Modeling</td><td><pre>transform Step<fire: in1.Event> from DFA
  out1.State(x) :- in1.State(x).
  out1.Event(x) :- in1.Event(x).
  out1.Transition(s, e, sp) :- in1.Trans
  out1.Current(sp) :- in1.Current(s), ir
  out1.Current(s) :- in1.Current(s), fai
}</pre></td><td>• Semantic Anchoring</td><td>• Semantics for complex DSMLs<br>• Composition</td></tr>
</table>

# Objectives

- A handy and intuitive modeling language
  - Constraining the Language
  - Incorporating existing semantic variations and standards (e.g. MAAB)
- Multiple Levels of Formality
  - Precise formal specification
  - Intuitive, annotated and excerpted formal specification
- Verified Formal Specification
  - Testing executable specifications
  - Bounded model checking on specifications
- Supporting Iterative Development Model
  - Regular updates, revisions, and extensions to the integration language

# Semantic Anchoring Dissected



# Examples

- A handy and intuitive modeling language
  - MAAB
- Multiple Levels of Formality: Electrical Power Port
  - HLE
  - HLE Explained
- Supporting Iterative Development Model
  - Metamodeling mathematics for denotational semantics (HLE, Stateflow)
  - Specifying operational semantics (Stateflow)
- Verified Formal Specification
  - Stateflow