Integration Platforms for the Model-Based Design of Cyber-Physical Systems

Janos Sztipanovits

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

Linkoping, SE
7 February 2018

Overview

- Introduction
- Horizontal Integration Platforms
- Component Modeling
- Automated Design Space Exploration
- Integration with Manufacturing
- Conclusion

CPS Time Line

- CPS was initiated in 2006/2007 by a group of academics, ex DARPA PMs with strong support from industry.

- CPS consolidates as a valid scientific direction by 2010. Universities led the charge, with VERY strong industry support. (Only complaint is the **CPS** name.)

- In 2010 NSF starts the CPS program in the Computer and Information Science and Engineering (CISE) directorate, establishes the **CPS Virtual Organization (CPS-VO.org)** at Vanderbilt-ISIS, starts **Annual CPS PI Meetings** . First ICCPS is in 2010 Stockholm. **DARPA starts the Adaptive Vehicle Make (AVM) program.**

- Between 2012-2015 industrial consortiums are created (**Industrial Internet Consortium (2014)**, **OpenFog Consortium (2015), IoT**, **Industry 4.0 (Germany)** and a "new Gold Rush" starts.

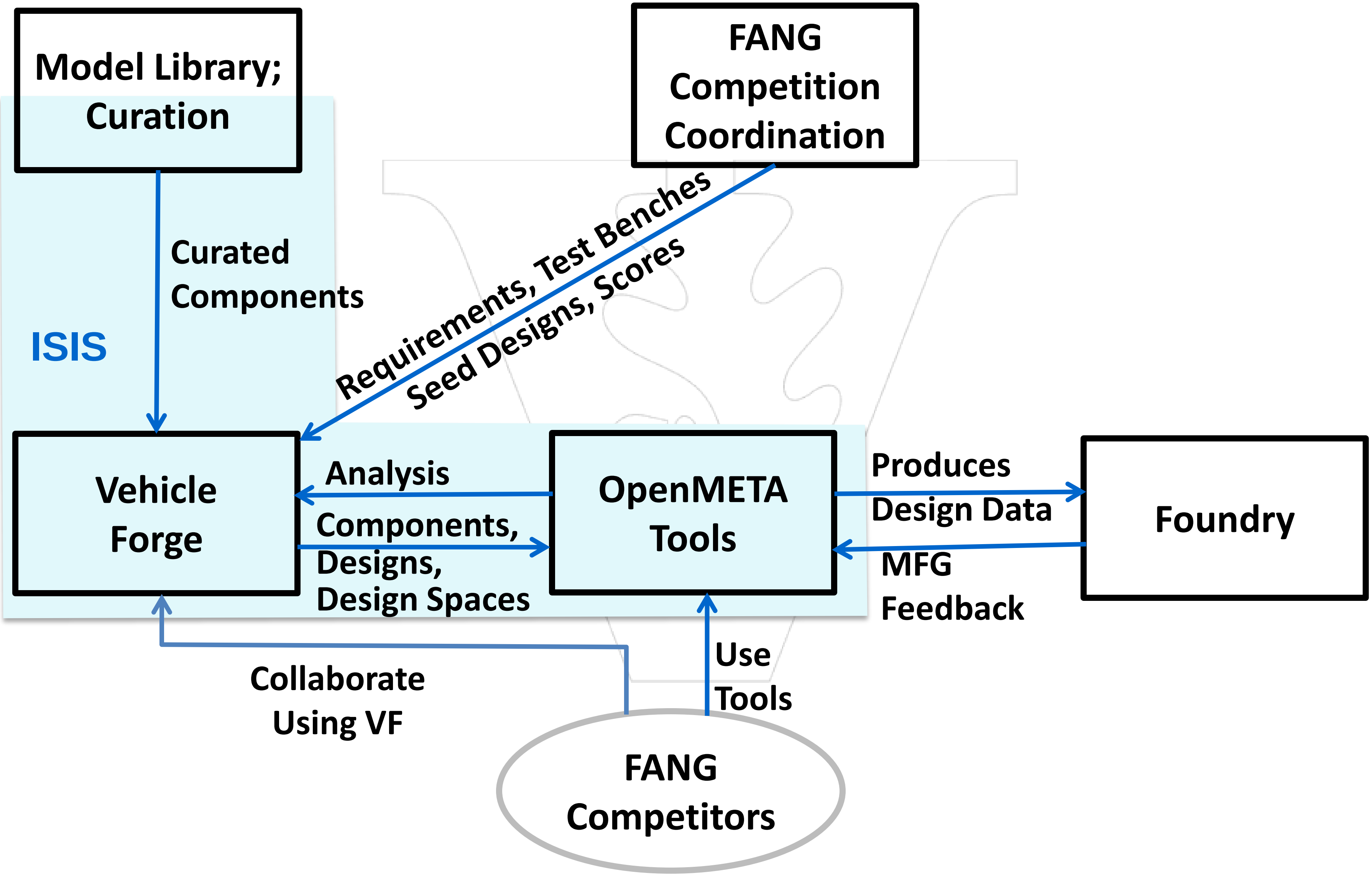
DARPA Adaptive Vehicle Make (AVM) Program

End-to-end model- and component-based design and integrated manufacturing of a new generation of amphibious infantry vehicle – a complex, real-life cyber-physical system. From infrastructure to manufactured vehicle prototype in five yeas (2010-2014).

Engineering/economic goals:

- **Shorten development time** by exploiting advantages of model and component based design
- Enable the adoption of **fabless design** and **foundry** concept in CPS: link design and manufacturing
- **“Democratize” design** by open source tool chain, crowed-sourced model library and prize-based design challenges

Major Components of AVM



Scientific Challenges

Achieving AVM goals require pushing the limits of "correct-by-construction" design using

– Model-based Technologies

Computational models that predict properties of cyber-physical systems "as designed" and "as built".

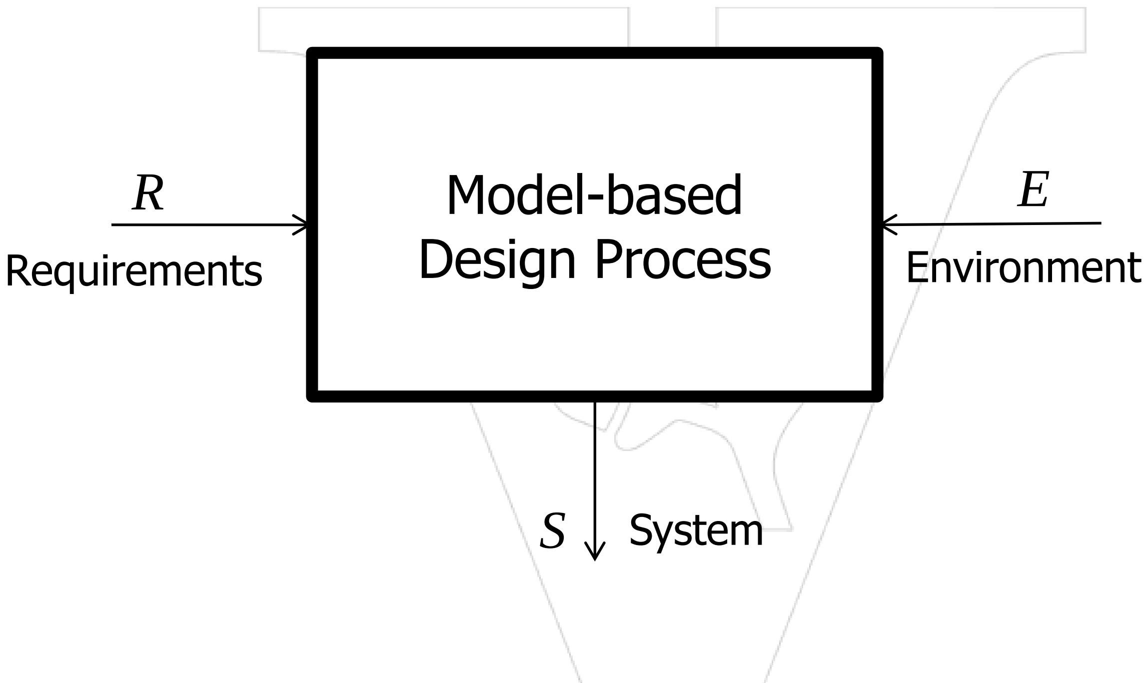
Challenge: Develop domain-specific abstraction layers for complex CPS that are evolvable, heterogeneous, yet semantically sound and supported by tools.

– Component-based Technologies

Reusable units of knowledge (models) and manufactured components.

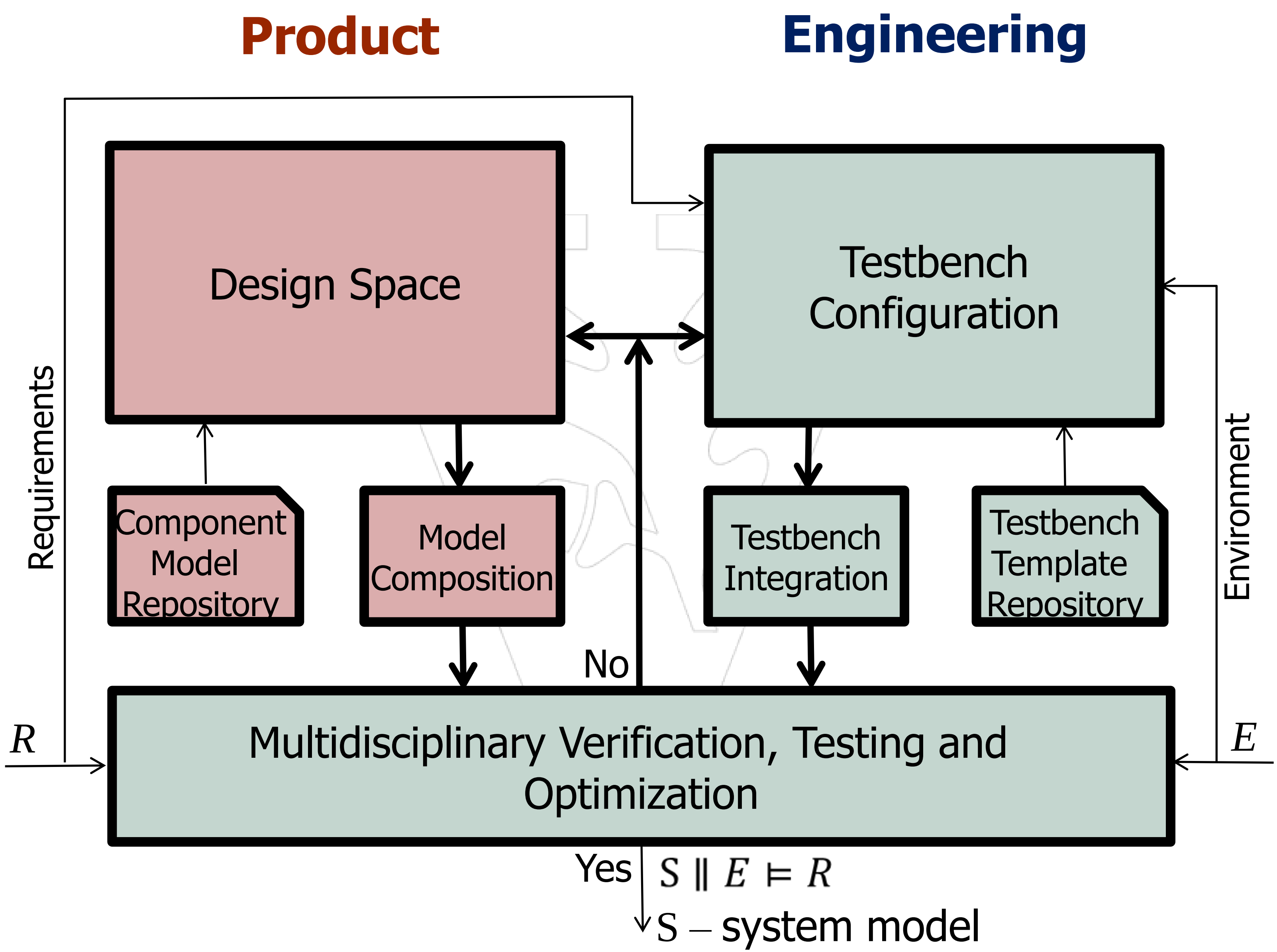
Challenge: Go beyond interoperability – find opportunities for composition where system-level properties can be computed from the properties of components

Model-Based Design



Design goal: $S \parallel E \vDash R$

Model- and Component-Based Design



Model- and Component-based Design Process

- Component Repository:

  $$C = \{C_i(x,p)\}$$

- For a system model *S:*

  $C_S = comptypes(S)$ denotes the set of component model types instantiated in *S*

  $C_S = comps(S)$ denotes the set of instantiated component models

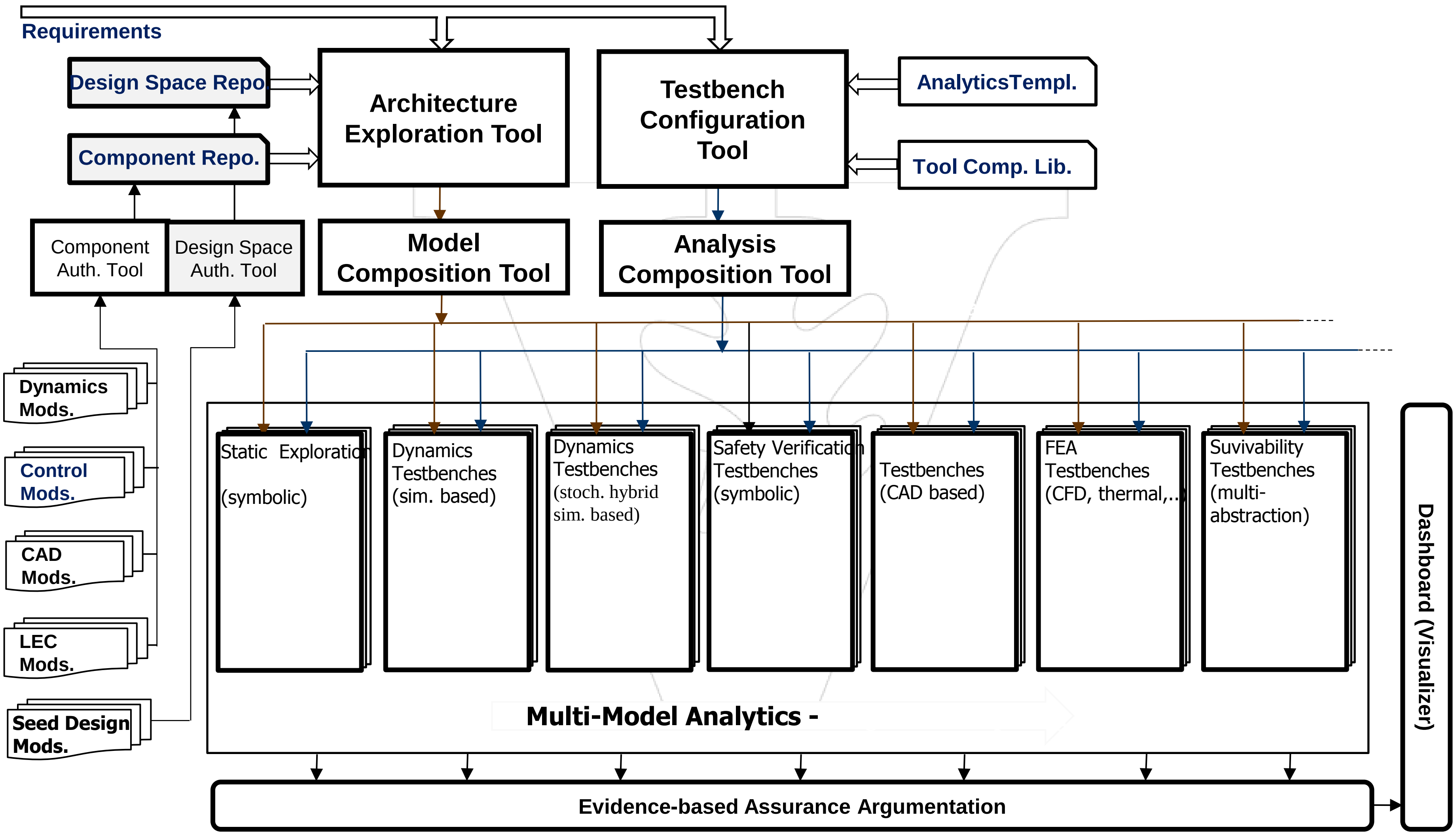
- The architecture of a system *S* is a labelled graph G_S, which is well formed if it satisfies a set of constraints Φ over G_S derived from the semantics of the interaction types

- The set of component types and composition constraints define a design space:

  $$D \stackrel{\text{def}}{=} \{S | G_S \vDash \Phi, comptypes(S) \subseteq C\}$$

- The goal of the design process is to synthesize $\boldsymbol{S \in D \text{ such that } S \parallel E \vDash R}$

Logical Architecture of OpenMeta

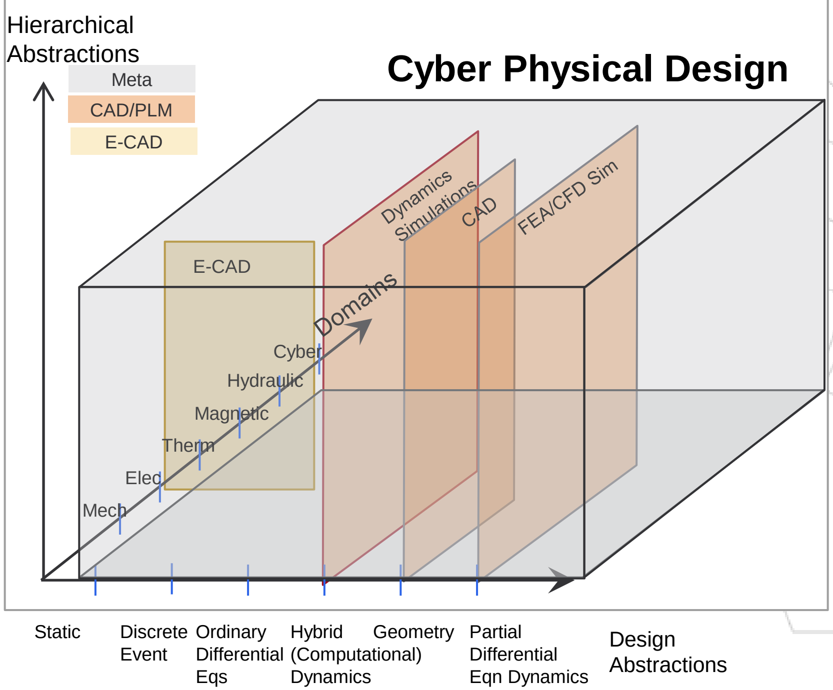


Overview

- Introduction
- ⇨ Horizontal Integration Platforms
- Component Modeling
- Automated Design Space Exploration
- Integration with Manufacturing
- Conclusion

CPS Design Domains and Tools

Heterogeneous Domains & Abstractions: **Model Integration**



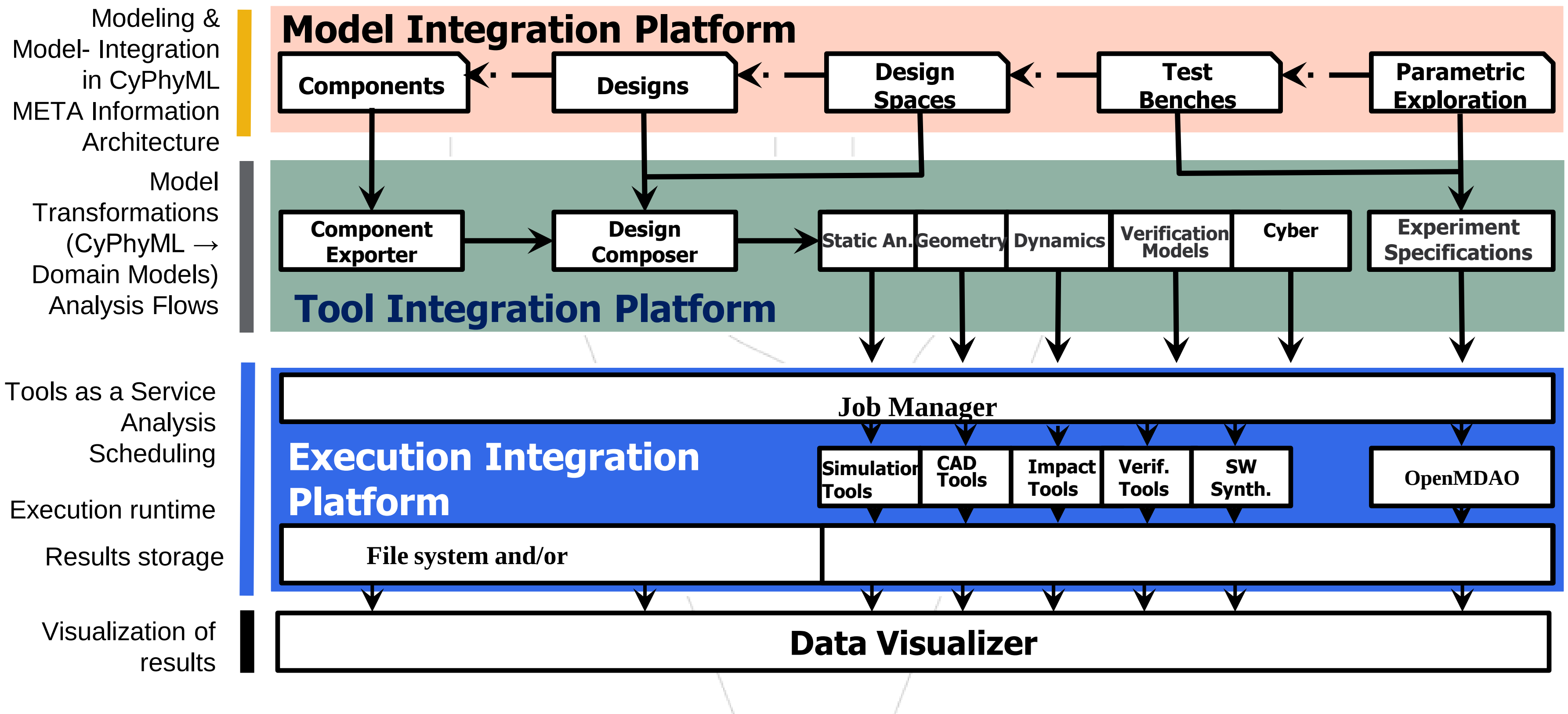
Heterogeneous Tools & Asset Libraries: **Tool Integration**

Integrated Engineering Tools

Lesson Learned -1: Need for Integration Platforms

- We found that the single most important change necessary to achieve correct-by-construction design is the introduction and systematic use of cross-domain modeling – consequently:
  - Vertically integrated tool suites should be complemented by horizontal integration platforms
  - Brings up interesting new challenges in modeling, tool architectures and deployment strategies

Result 1: OpenMETA Horizontal Integration Platforms

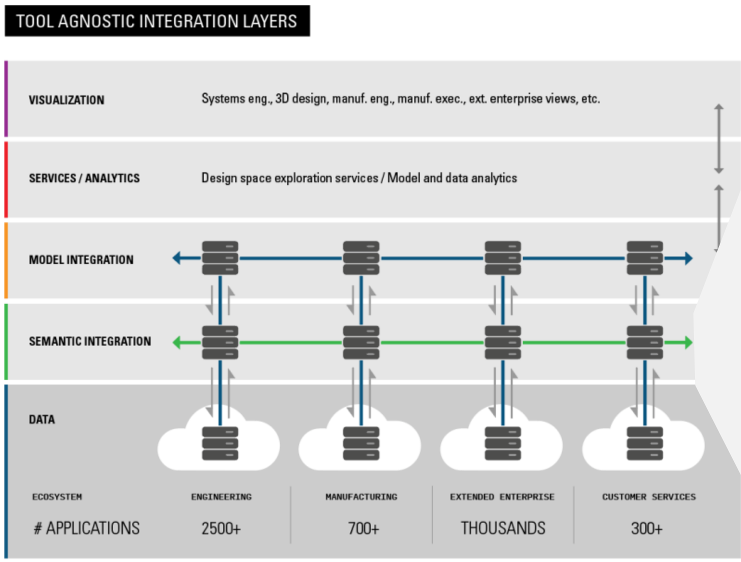


Horizontal Integration Platforms cut across traditionally isolated design domains.

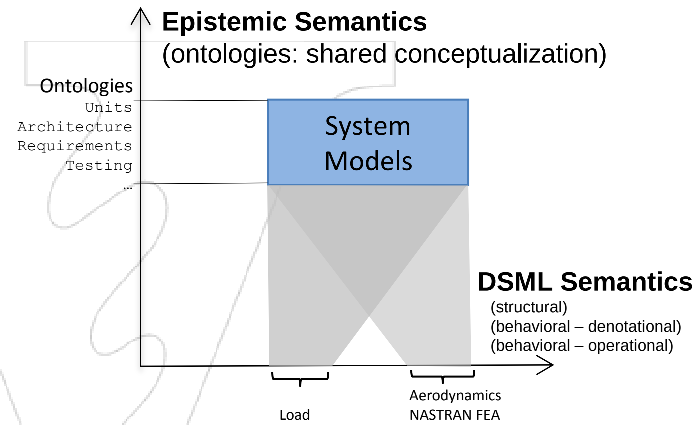
OpenMETA Horizontal Integration Platforms

- Model Integration Platform:
  Formal framework and tool for composing Model Integration Languages, metaprogrammable modeling tool, metamodel repository, Semantic Backplane
- Tool Integration Platform:
  Tools for specifying, implementing and composing model transformations, platforms for orchestrating tool execution in design flows
- Execution Integration Platform:
  Cloud-based deployment infrastructure, web-based delivery of design tools, data repositories and visualizers

Semantic Integration and Model Integration



Dimensions of Semantics



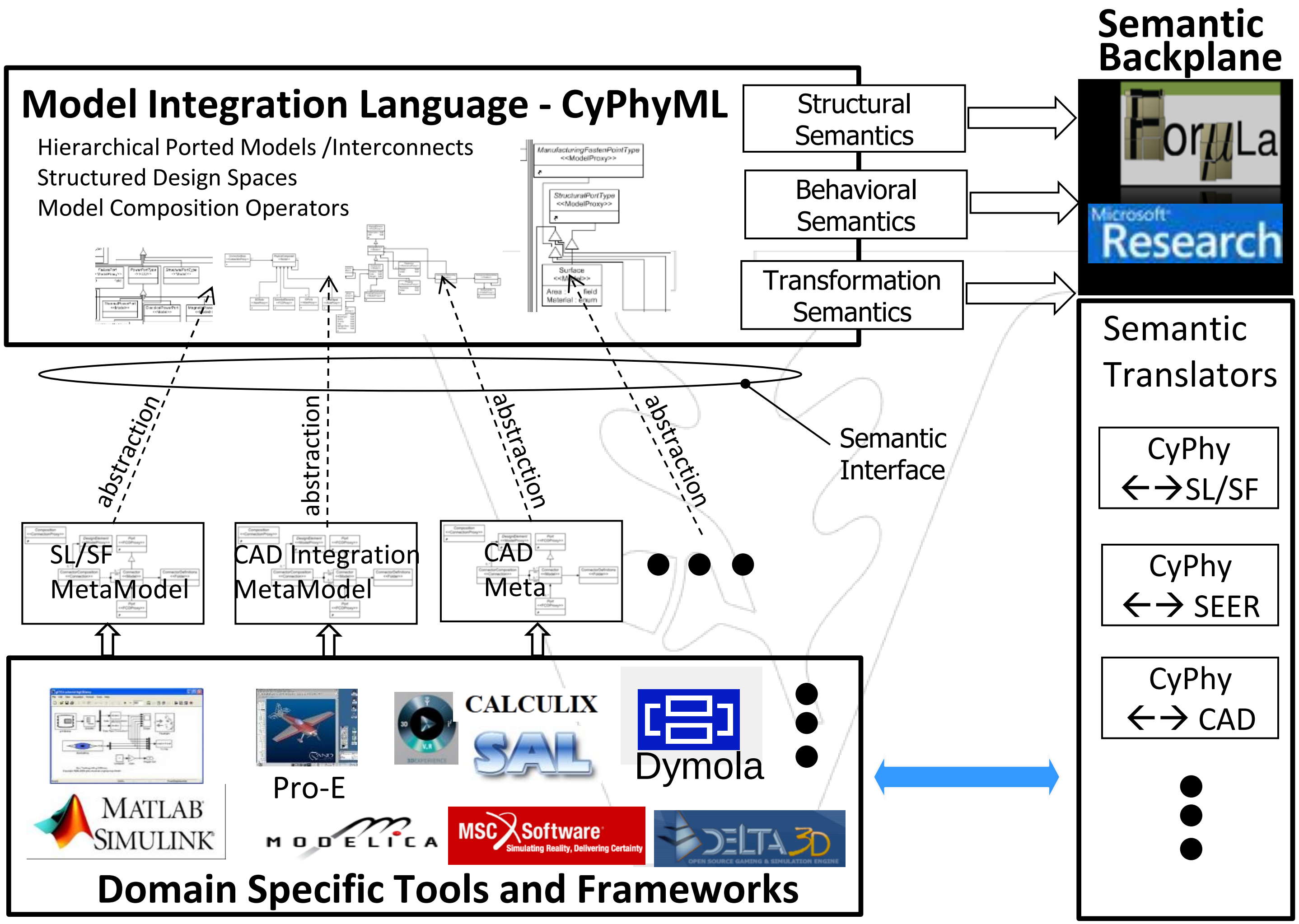
Semantic Integration:

- Data is tagged with metadata
- Metadata is structured by ontologies
- DSMLs are defined driven by needs of analyses
- Semantics of relationships among DSMLs is formally specified

Model Integration:

- Models are built using DSMLs
- Model composers are developed
- Model transformation tools for analytics are created using DSML semantics
- Multi-models are created using Model Integration Languages

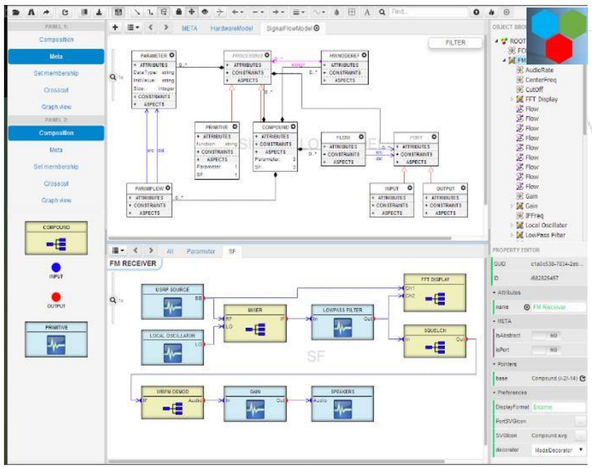
Result 2: Semantic Integration



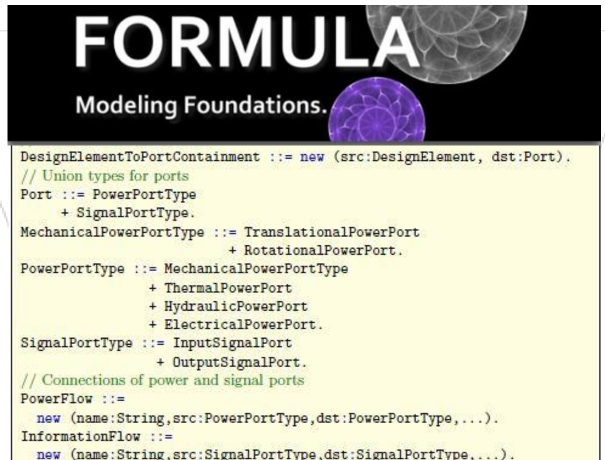
Impact: Open Language Engineering Environment → Adaptability of Process/Design Flow → Accommodate New Tools/Frameworks , Accommodate New Languages

Result 3: Semantic Backplane

Vanderbilt
https://github.com/webgme



Microsoft Research
github.com/Microsoft/formula



Microsoft Research
https://github.com/Z3Prover/z3

Z3

Metamodels
Model transformations
Repositories

Metamodel Specs
Transformation Specs
Semantic Specs

Solvers
Domain Theories
Model Finder

Overview

- Introduction
- Horizontal Integration Platforms
- ⇨ Component Modeling
- Automated Design Space Exploration
- Integration with Manufacturing
- Conclusion

Lesson Learned – 2: Need for Component Modeling Technology

- Rich interfaces decoupled from modeling languages used for capturing domain models
- Compositionality and semantically well defined composition operators
- Explicit bounds for composability
- Inclusion of relevant suite of domain models – on multiple levels of fidelity
- Documented validity

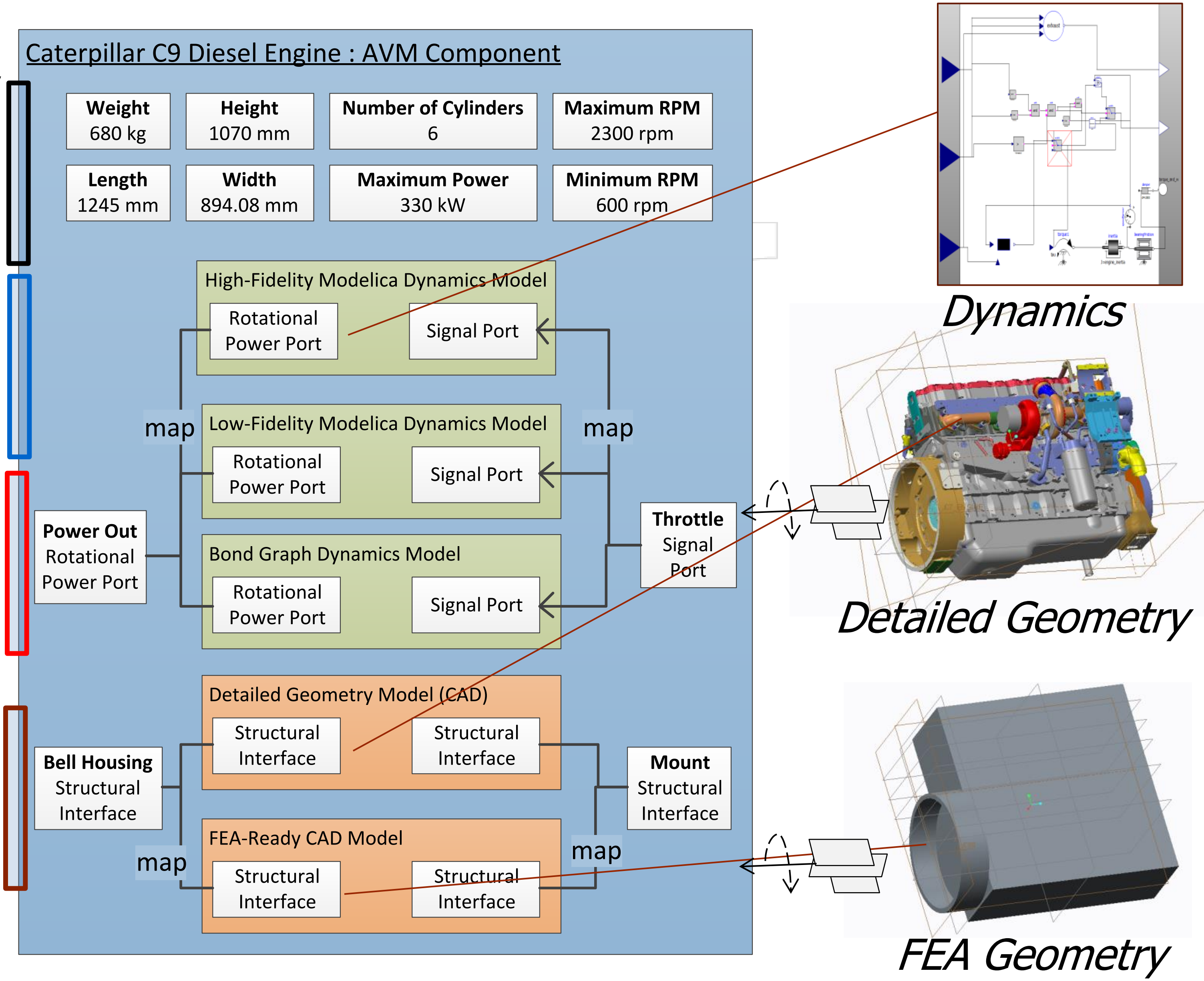
AVM Component Model

Param./Property Interfaces
- characterize
- configure

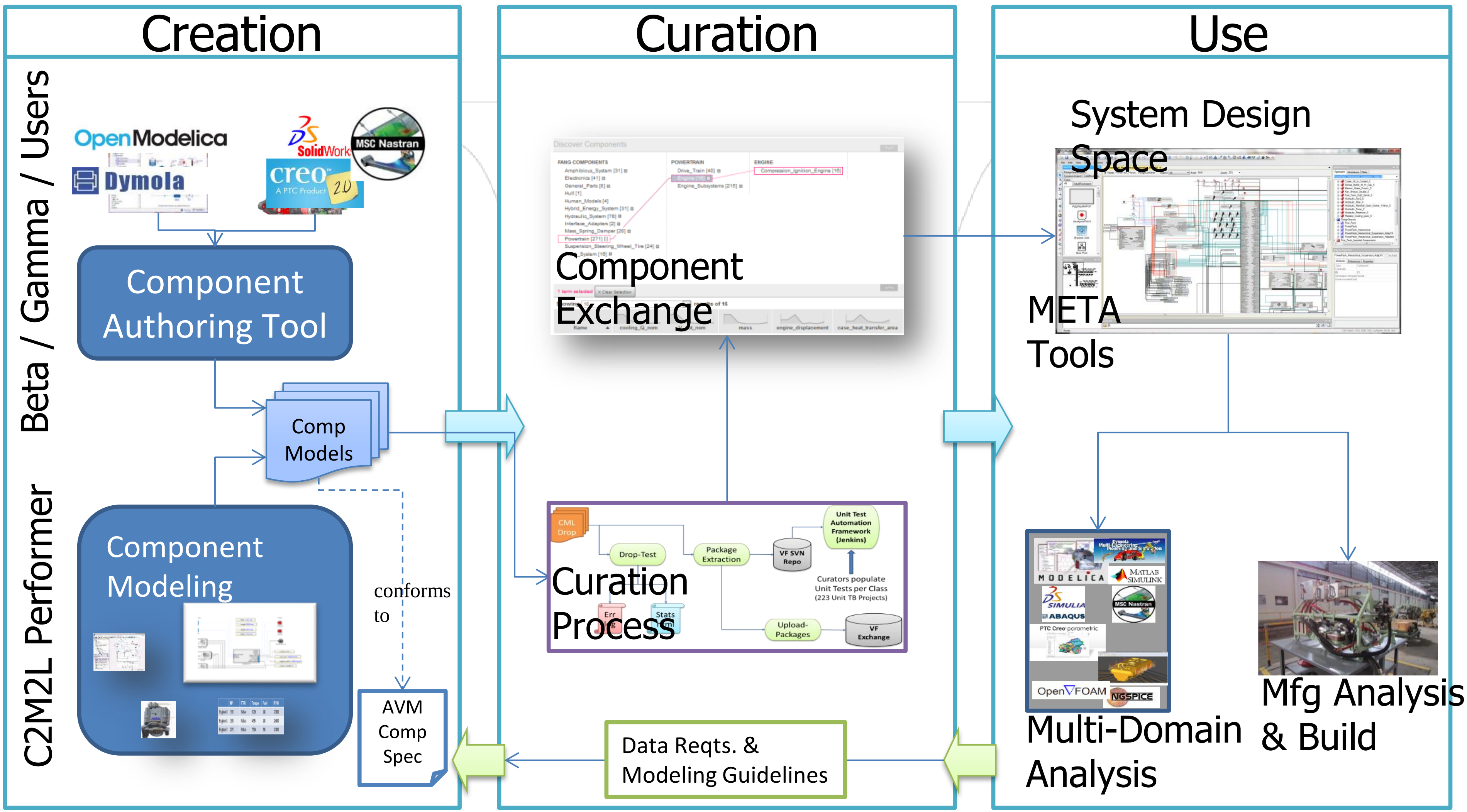
Signal Interfaces
- causal/directional
- logical conn.
- no power transfer

Power Interfaces
- acausal
- physical phen. (torque/angle..)
- power flow

Structural Interfaces
- named datums
- surface/axis/point
- mapped to CAD



Making AVM Components



Experience

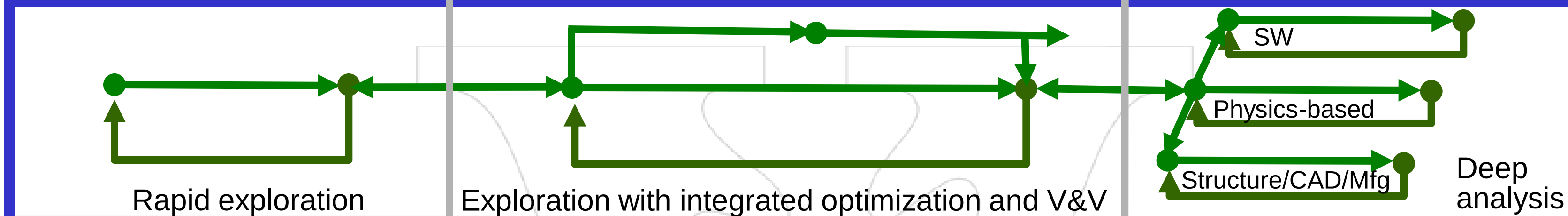
- Building *good* model libraries is hard
- Component models should not be confused by sub-models taken from an existing complete design: component models need to be flexible and remain composable in many designs
- **Regions of validity** (i.e. composability) need to be explicitly represented
- **Parametric uncertainties** need to be explicitly represented
- **Epistemic uncertainties** need to be assessed

Overview

- Introduction
- Horizontal Integration Platforms
- Component Modeling
- ⇒ Automated Design Space Exploration
- Integration with Manufacturing
- Conclusion

Lessons Learned – 3: Need for Automation in Design Flow

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



Architecture Design

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

Integrated Multi-physics/Cyber Design

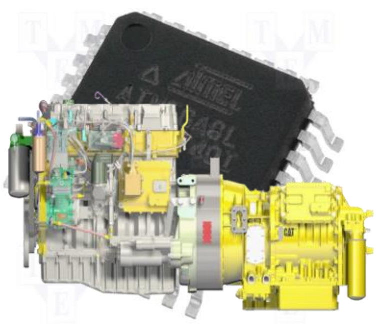
- Design Space + Behavioral Constraint Modeling
- Architecture Modeling
- Dynamics Modeling (multiple abstractions and multiphysics)
- CAD/assembly modeling
- Coarse Manufacturing Constraint Modeling

Detailed Design

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

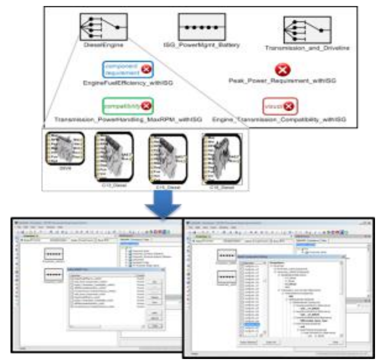
Result 4: Design-Space Exploration Using Progressive Refinement

Component Model Repositories (Executable Descriptors)



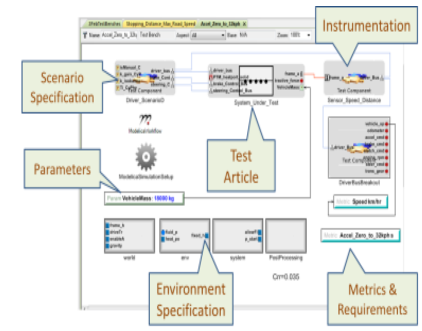
- Design Reuse
- Leverage cross-domain Component Libraries

Composed System Model for Design Space Exploration (using progressive refinement)



- Traverse design space, evaluate, understand tradeoffs
- Maintain design flexibility & agility

Executable Requirements (virtual test benches)

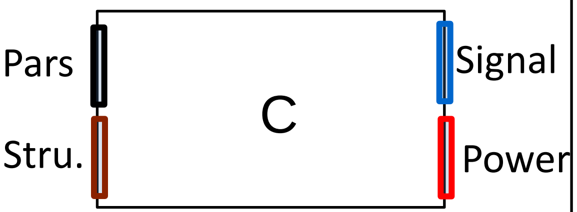
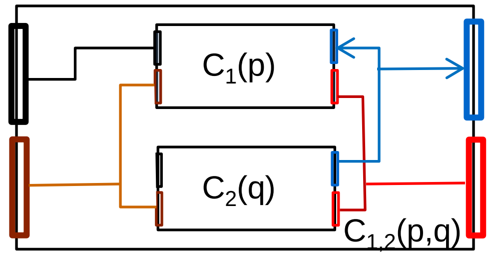
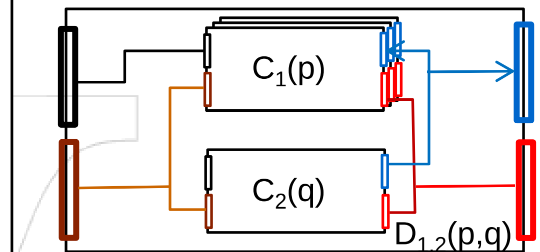


- Auto-compose models & invoke domain analysis tools
- Minimize repetitive engineering labor

Auto-Composition

MODELICA | NGSPICE | MATLAB SIMULINK | creo™ A PTC Product | SIMULIA ABAQUS | Adams | SYSTEM C™ | OpenFOAM

Components, Designs, Design Spaces

Components	Designs	Design Spaces
		
Self-contained building block	**Instantiate** and **connect** Components	Sets of **parameterized architectures**
Properties and **Parameters**	Parameters, behaviors, geometry are **composed**	Extended around **seed designs**
Wrapper for **detailed domain models**	Can be wrapped as a **component**	Shaped by **design and manufacturability constraints**
Aggregates the domain interfaces into a **single set of component interfaces**	Aggregates the component interfaces into a **single set of system interfaces.**	Accumulates, evolves **design and manufacturing knowledge**

Design Space Construction and Exploration

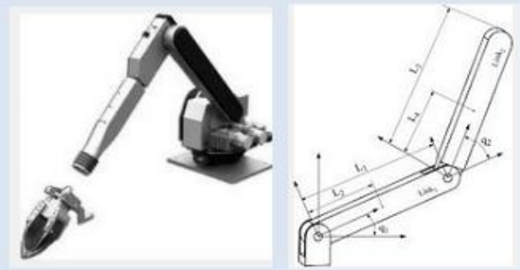
Architectures:

Mobility
- # Wheels/Legs/Tracks
  - Wheel Articulation

Power Sources
- Diesel, Electric, S

Manipulators
- ...

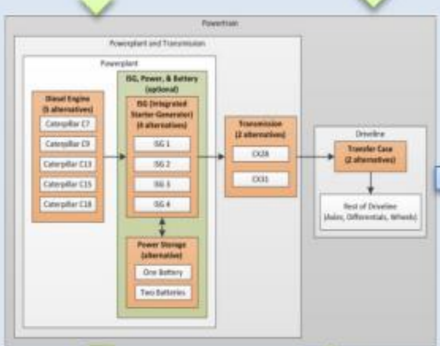
Parametric Variation:



Component Choices:

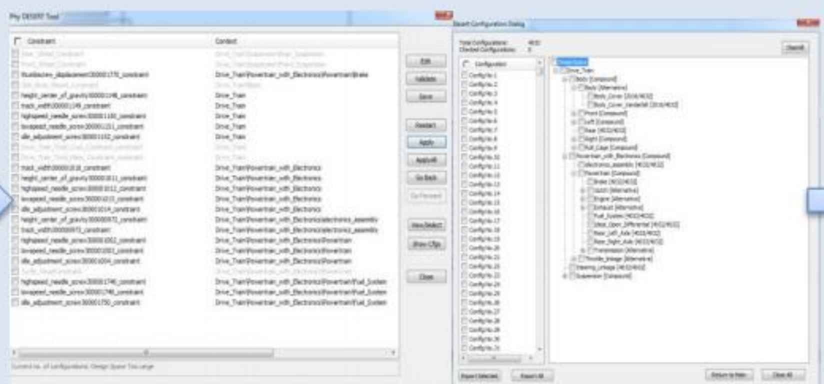
Components

Subsystem Design Spaces

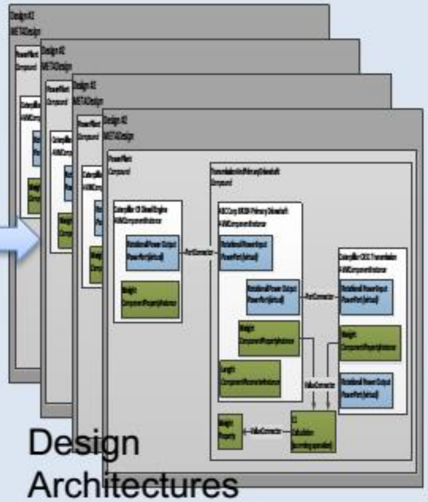


Design Space Evolution

Constraint-guided Design Space Pruning



BDD Based Constraint Solver, Scales to Large Design Spaces

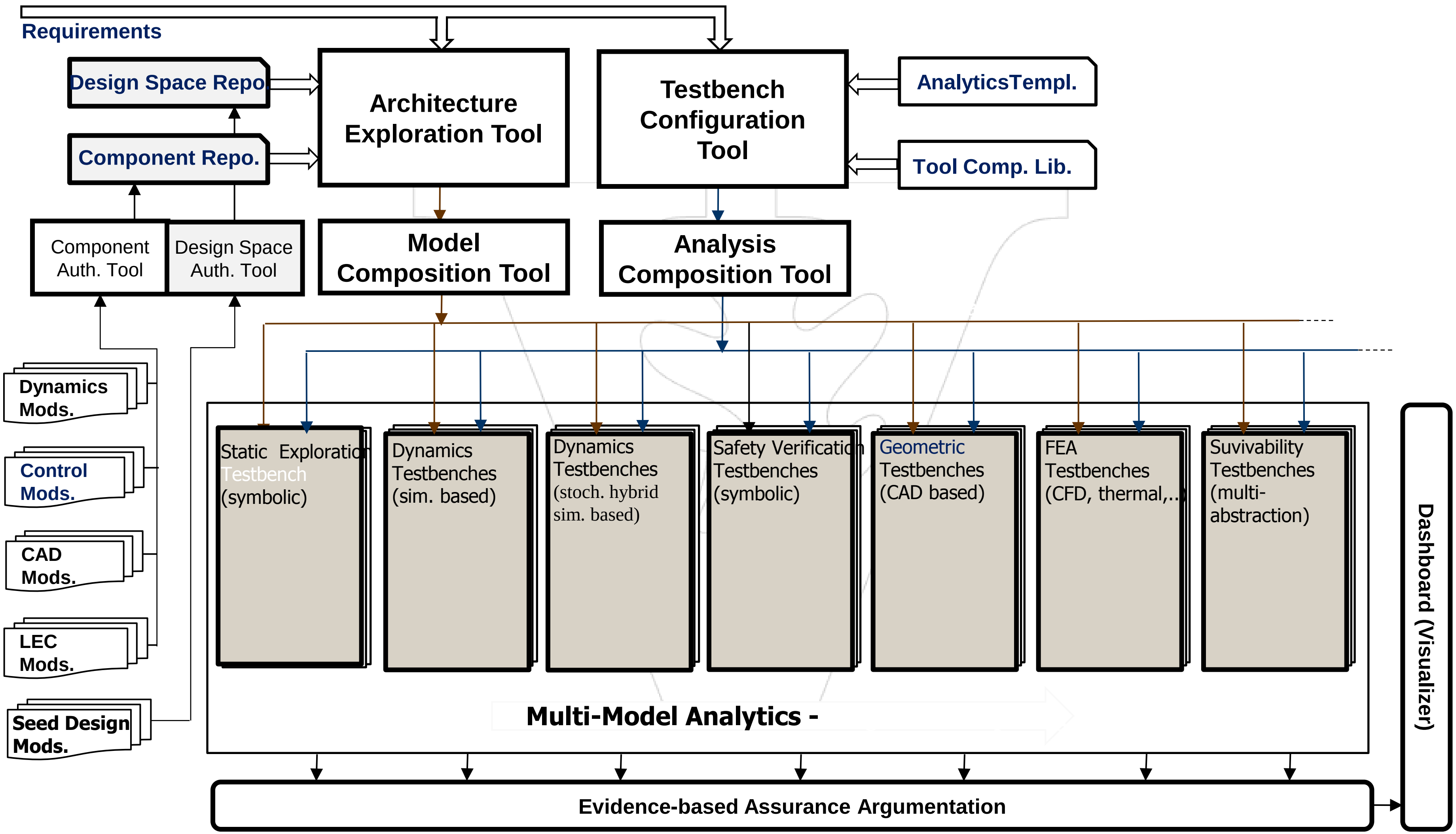


Design Architectures

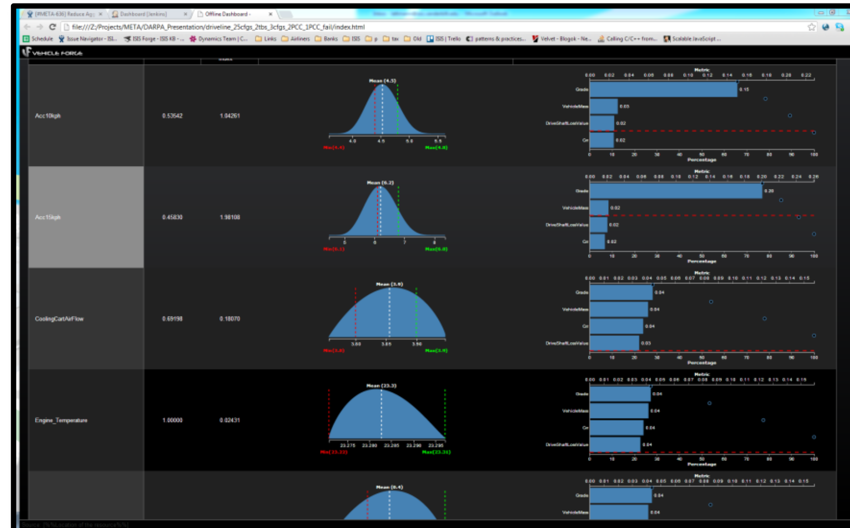
Experience

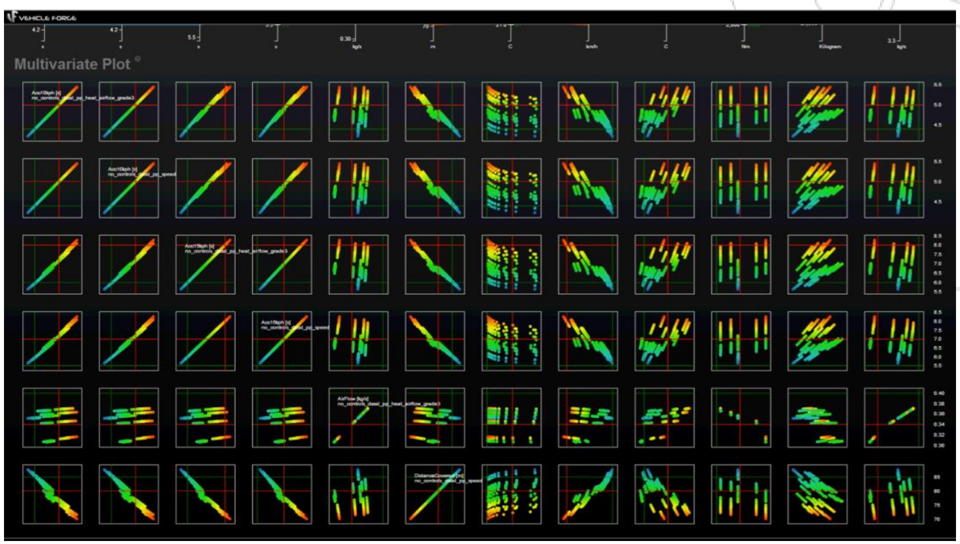
- **Design spaces need to be constructed** to make design space exploration meaningful, tractable
- **Seed designs** have significant importance in constructing meaningful design spaces
- Information management infrastructure need to **follow changes**: design spaces can expand/shrink with technology changes and knowledge about regions in the design space can increase

Result 5: Automated Analysis Using Virtual Test Benches



Result 6: Design Space Analyzer and Visualizer



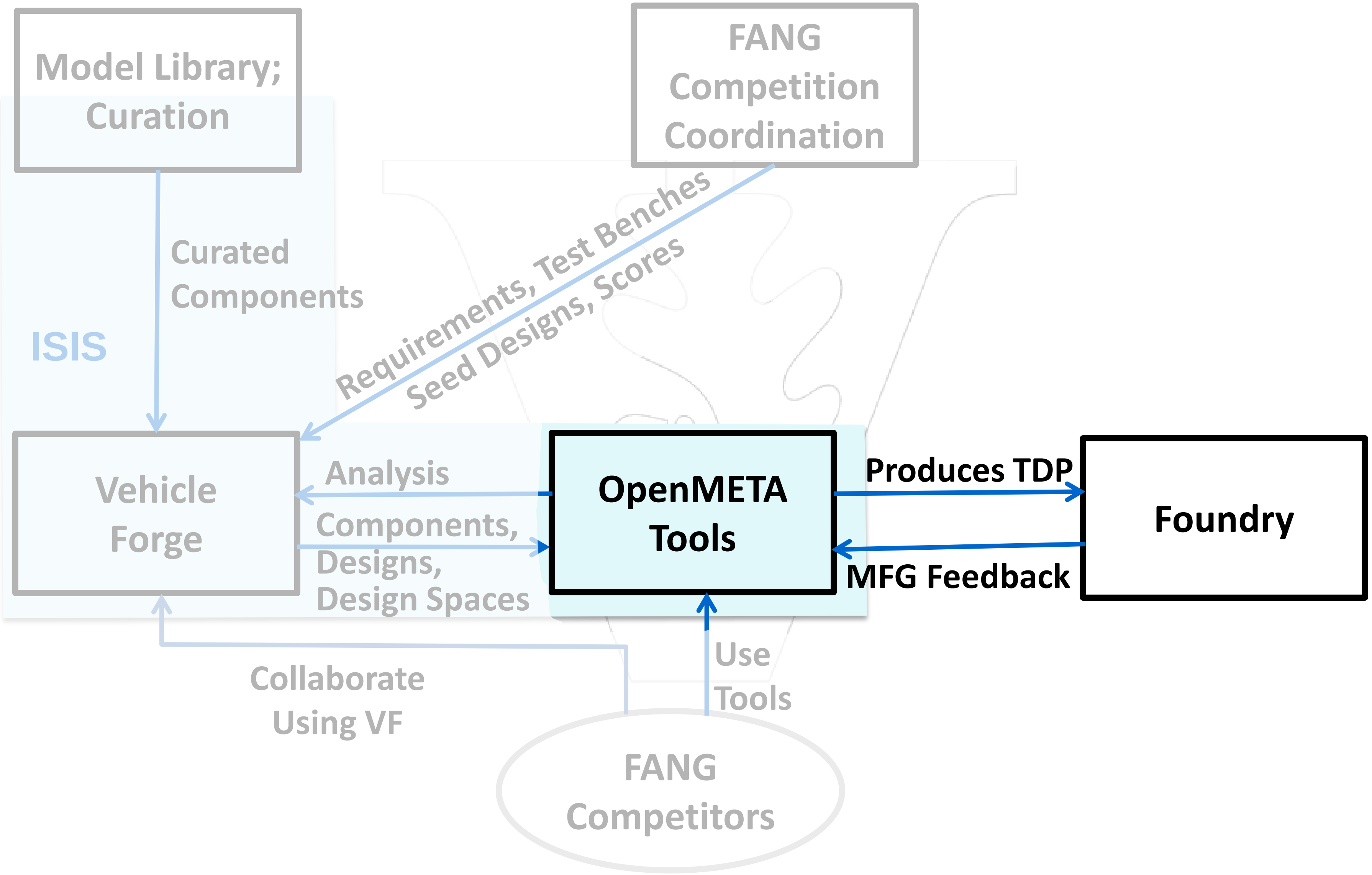


Interactive tools for multi-objective optimization (Georgia Tech team)

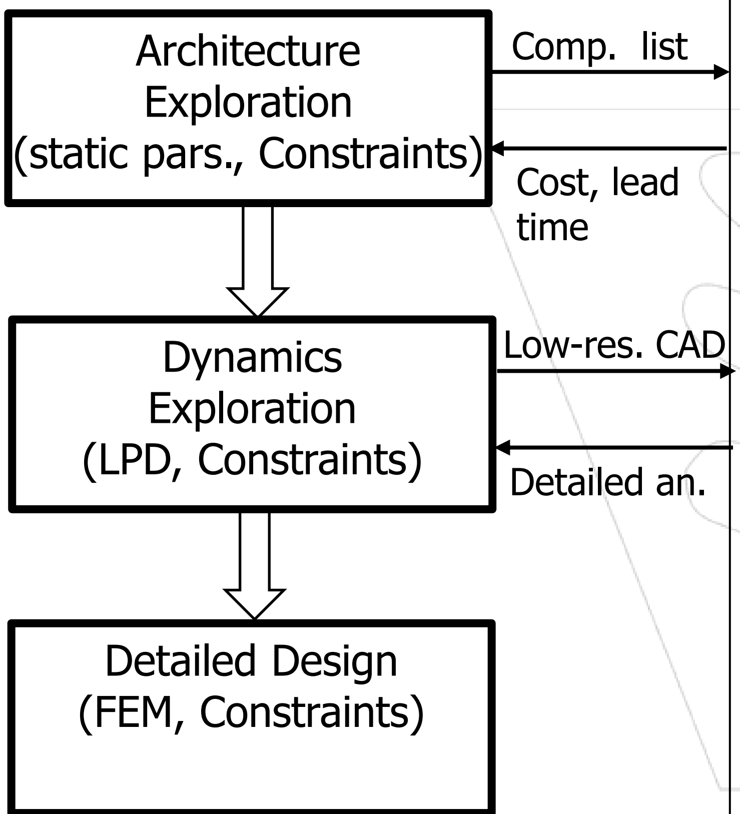
Overview

- Introduction
- Horizontal Integration Platforms
- Component Modeling
- Automated Design Space Exploration
- ⇨ Integration with Manufacturing
- Conclusion

Lessons Learned – 4: Need for Interface Between Design and Mfg.



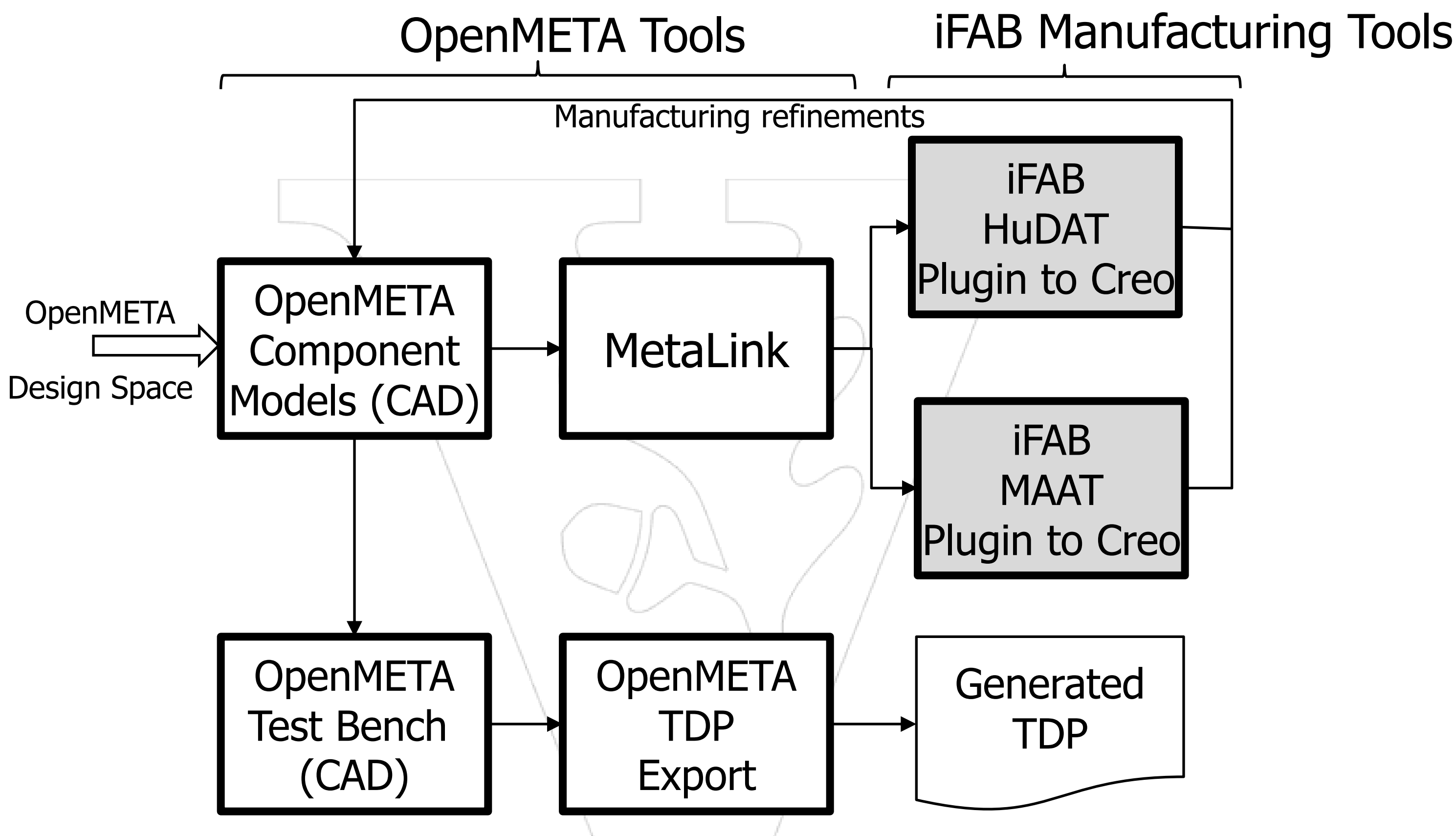
Result 7: Design Space Exploration with Manufacturability Test



- Manufacturing Feedback Analysis
  - Cost, lead time
  - Conceptual Analysis (completeness)
  - Detailed Analyis (machinability, eq. availability)
  - Reliability, Availability, Maintainability, Durability (RAMD)

Abstraction mismatch

Refined Manufacturing Interface



MetaLink: Architecture level synchronization between OpenMETA and Creo

Experience

- Manufacturability parameters even early design space exploration help… (lead time, cost)
- Linking manufacturing refinement tools with architecture modeling tools was quite effective
- Open path toward product and manufacturing process co-design

OpenMeta Under Transitioning

- CyPhy Language & Infrastructure
  - Extend Components, Designs, Design Spaces
  - Other Tool Integration Into Design Flow
- Design Space Exploration & Visualization
  - Apply Constraints, Explore Design Options
- Model Composers and Test Bench Analyses:
  - Modelica, CAD, Simple FEA/CFD Composition, Blast & Ballistics, Manufacturability
- Google's first purchase order led to the foundation of MetaMorph Inc. – spinoff from Vanderbilt-ISIS

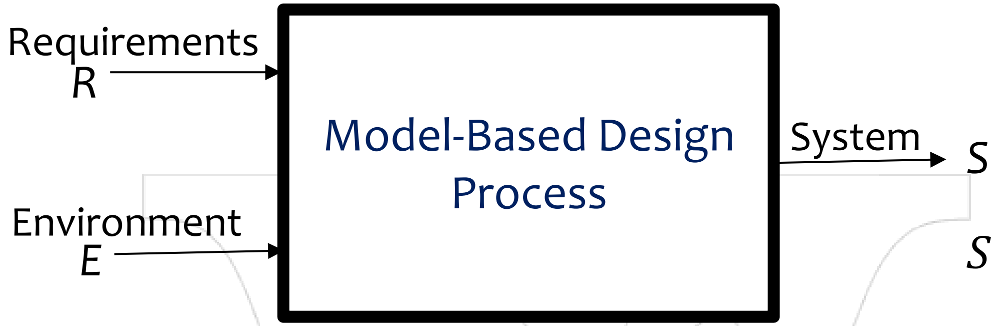
Ongoing Research

- Product and manufacturing process co-design
  - Rapid Feedback On Manufacturing Decisions
  - Joint Architecture/Manufacturing Process Design
- Increased Control in PDE Analysis
  - Meshing, Forces & Constraints
- Multi-User Concurrent Design
- Tools Integration
  - SPICE, SystemC, Aero, Space, Uncertainty Management
- Pilots in different domains.

Conclusion

- Horizontal Integration Platforms
  – Infrastructure is reusable in many domains
- Semantic Integration
  – Model Integration Languages, explicit semantics
- Increased Automation
  – Design space exploration using progressive refinement
- Product and Manufacturing Process Co-design
  – Next revolution?

Model-based Design and Societal-Scale CPS



$S \parallel E \vDash R$

The goal of model-based design is to synthesize S from a class of systems $\mathbb{C}_S$ such that $S \parallel E \vDash R$.

SID methodology for formal verification (Seshia, 2015):
- Abstraction-Based Model Checking
- Synthesis of *R* (STL formula) from sim. traces..

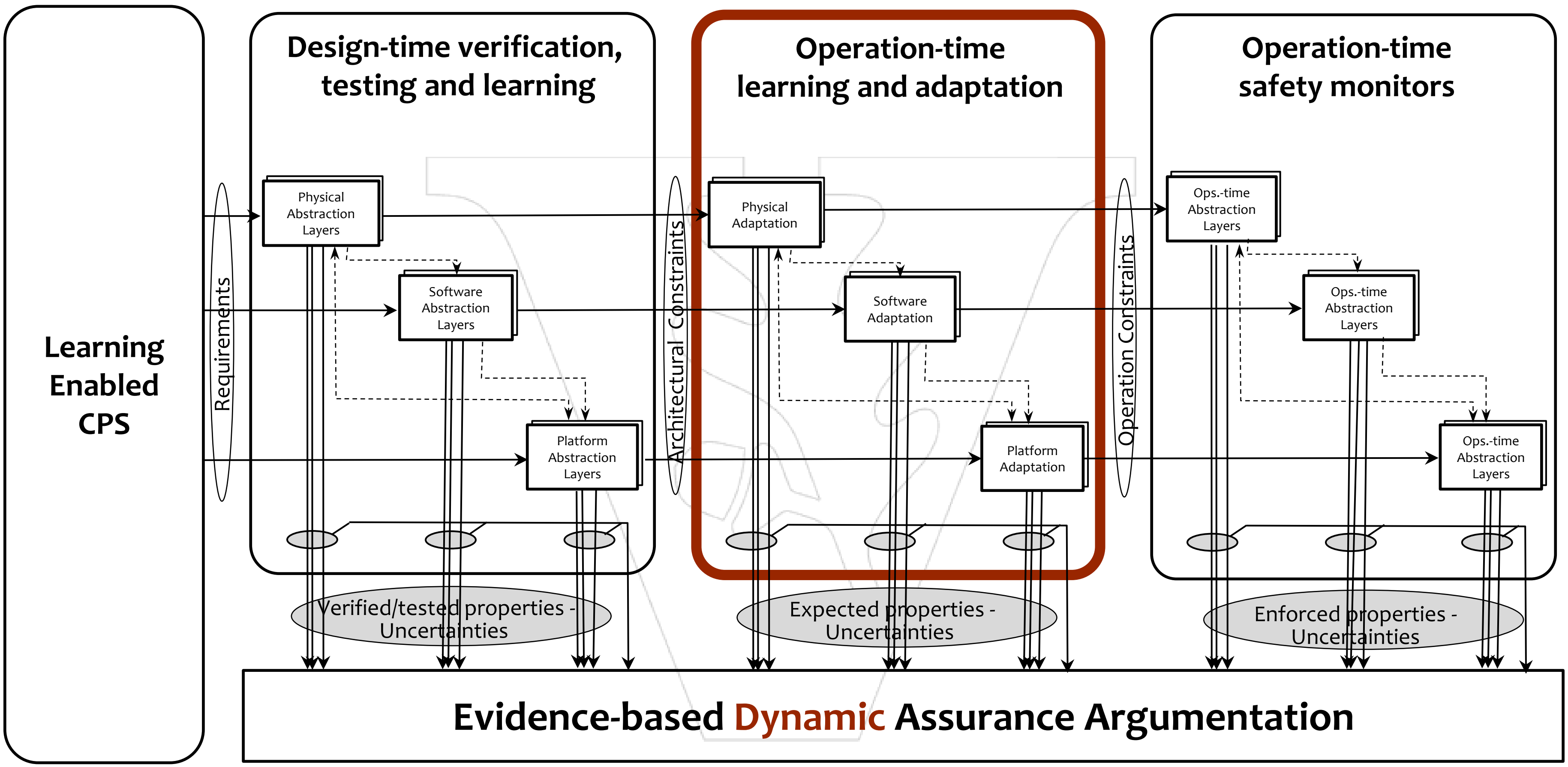
Scalability remains limited for CPS.

Modeling uncertainty in physical systems
- **Aleatoric** uncertainties: irreducable, rooted in physics
- **Epistemic** uncertainties: lack of knowledge

Role of epistemic uncertainties dominates.

Addressing Epistemic Uncertainties with Learning

Extending OpenMeta with Learning Enabled Components:



Assurance using design-time (partial) evidence

Assurance using operation-time evidence

Assurance using operation-time observations

DARPA: Assured Autonomy Program

Current Trends

Industry Perception:
(Gartner's View on Technology Trends 2016)

- **Transparently immersive experiences**
  Technology becomes more adaptive, contextual and fluid
- **The perceptual smart machine age**
  CPS fusion with AI
- **Platform revolution**
  Ecosystem-enabling platforms

Academic Perception (Current Academic Research Trends)

- **Policy awareness**
  How to build H-CPS that can be parameterized with societal context?
- **Learning Enabled Components**
  How to deliver assurance?
- **Platforms with safety, security and performance guarantees**

Future

- The next decade is (probably) about
  - The merge of AI and CPS
  - Spread of societal-scale CPS
  - Autonomy everywhere

- Threats
  - Can societies follow the speed of change?
  - Can we make a dent on the security problems?
  - Can we keep the future systems assured?