Georgia Institute of Technology

Systems Realization Laboratory

Model-Based Systems Engineering

Chris Paredis

Systems Realization Laboratory

Product and Systems Lifecycle Management Center

G.W. Woodruff School of Mechanical Engineering

Georgia Institute of Technology

www.srl.gatech.edu www.pslm.gatech.edu

© 2008, Chris Paredis

Presentation Overview

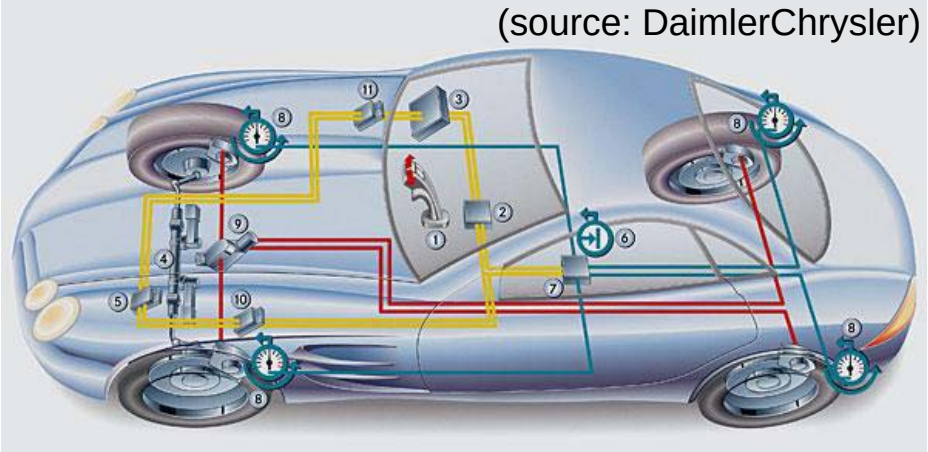
- Context: What is Model-Based Systems Engineering? Why MBSE?
- A simple model of knowledge and information in MBSE
- Which kinds of knowledge and information?
  - Information schema
  - Language mappings
  - Abstractions and idealizations
  - Analysis patterns
  - Synthesis patterns
  - Design Workflow
- Challenges and Summary

© 2008, Chris Paredis

A Systems Approach for Product Development

- Technology Trends
  - Miniaturization
  - Embedded intelligence
  - Networked connectivity
- Modularization
  - One subsystem per function
  - Standardized interfaces
- Systems Approach
  - Increased number of functions
  - New functions by integrating multiple subsystems
  - New architectures

(source: DaimlerChrysler)

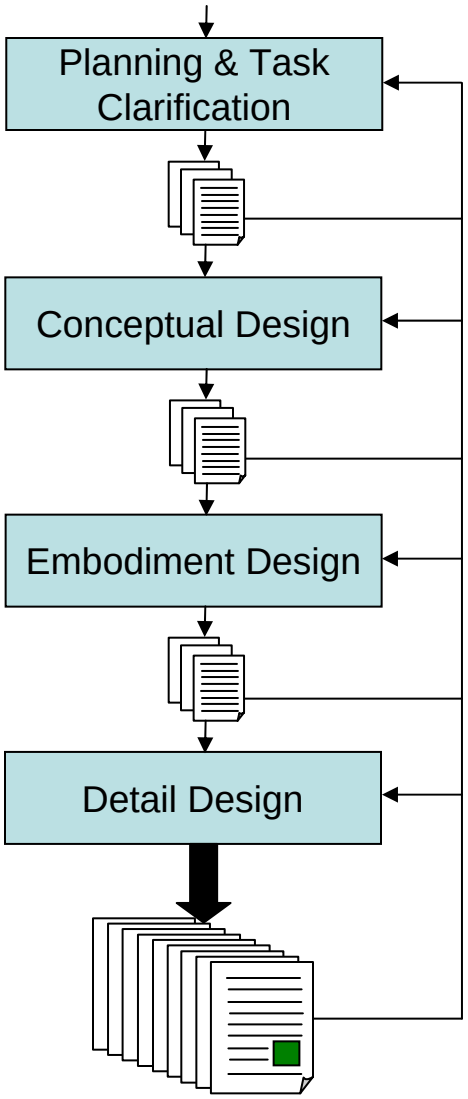


© 2008, Chris Paredis

Traditional Design Approaches

- Document-centric, systematic information transformations
- Generally, information is transferred manually between design steps and team members
- Output is large set of documents

→ Inadequate for contemporary SE



© 2008, Chris Paredis

Challenges in Systems Engineering

- Multiple integrated functions
- Multiple engineering disciplines
- Multiple stakeholders
- Globally distributed, heterogeneous design teams
- Complex, emergent system behavior
- Large quantities of design knowledge and information

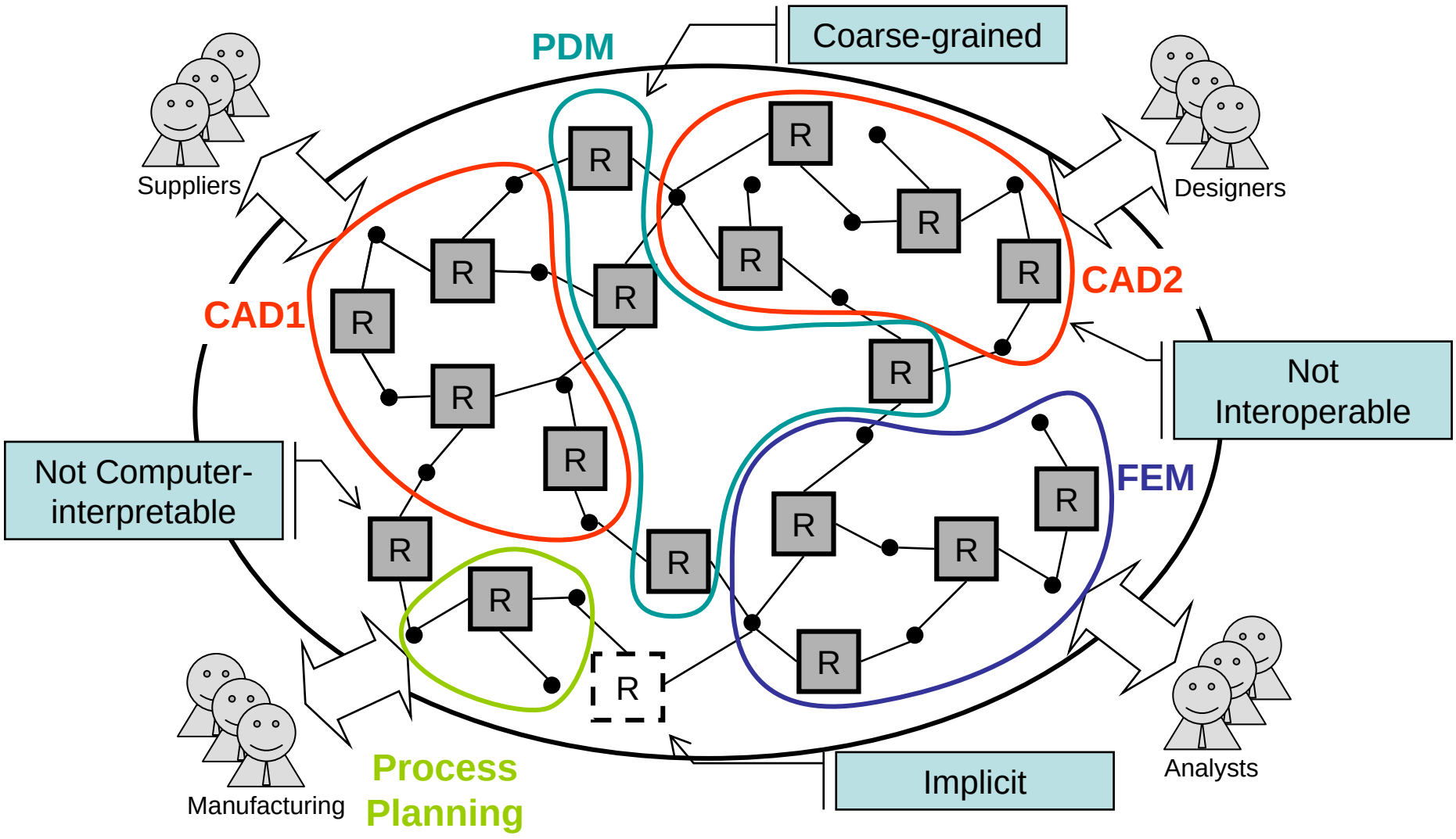
→ Need Formal, Model-Based Approach

© 2008, Chris Paredis

Presentation Overview

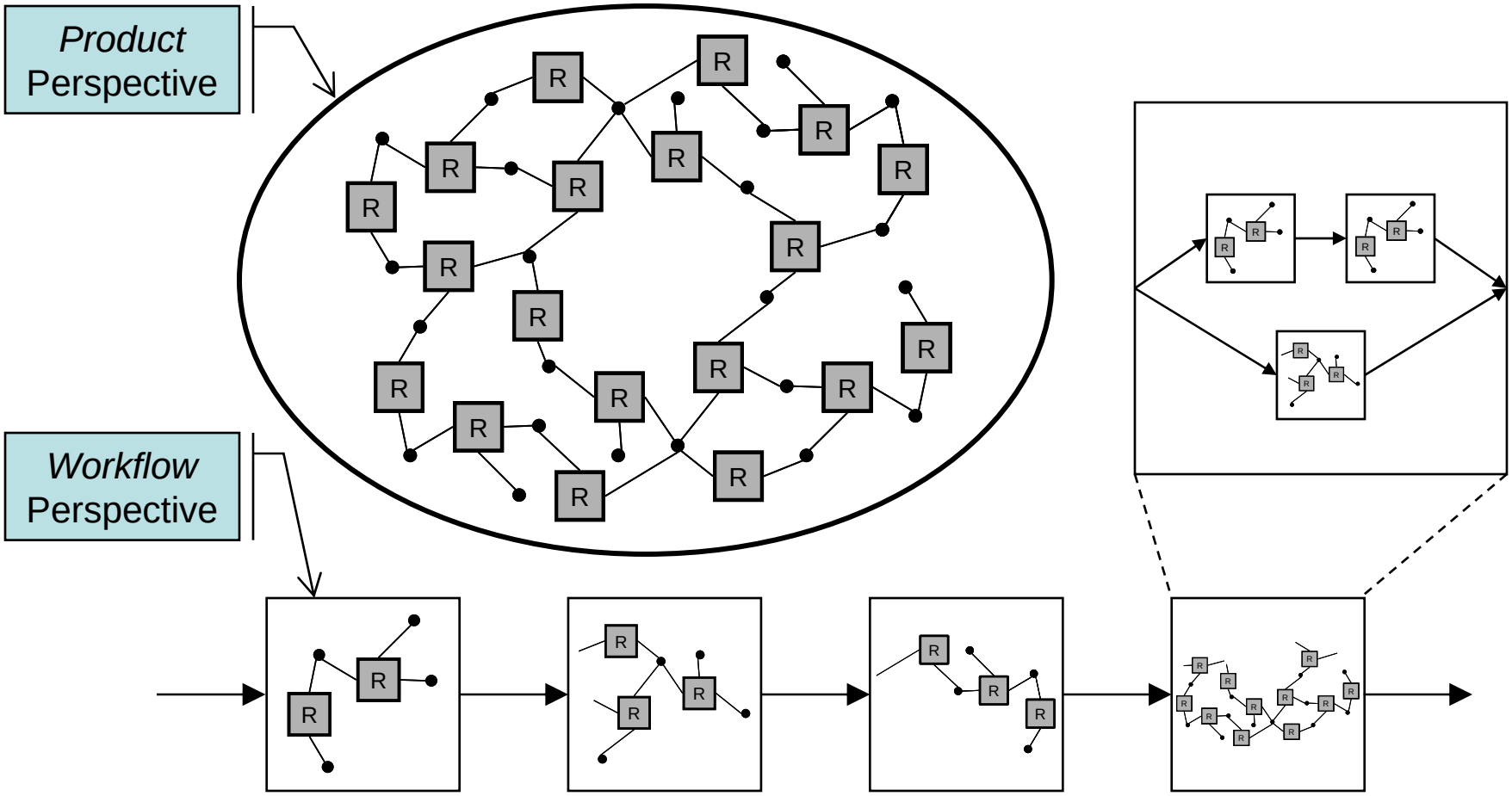
- What is Model-Based Systems Engineering?
  Why MBSE?
- A simple model of knowledge and information in MBSE
- Which kinds of knowledge and information?
  - Information schema
  - Language mappings
  - Abstractions and idealizations
  - Analysis patterns
  - Synthesis patterns
  - Design Workflow
- Challenges and Summary

Knowledge & Information in Systems Engineering



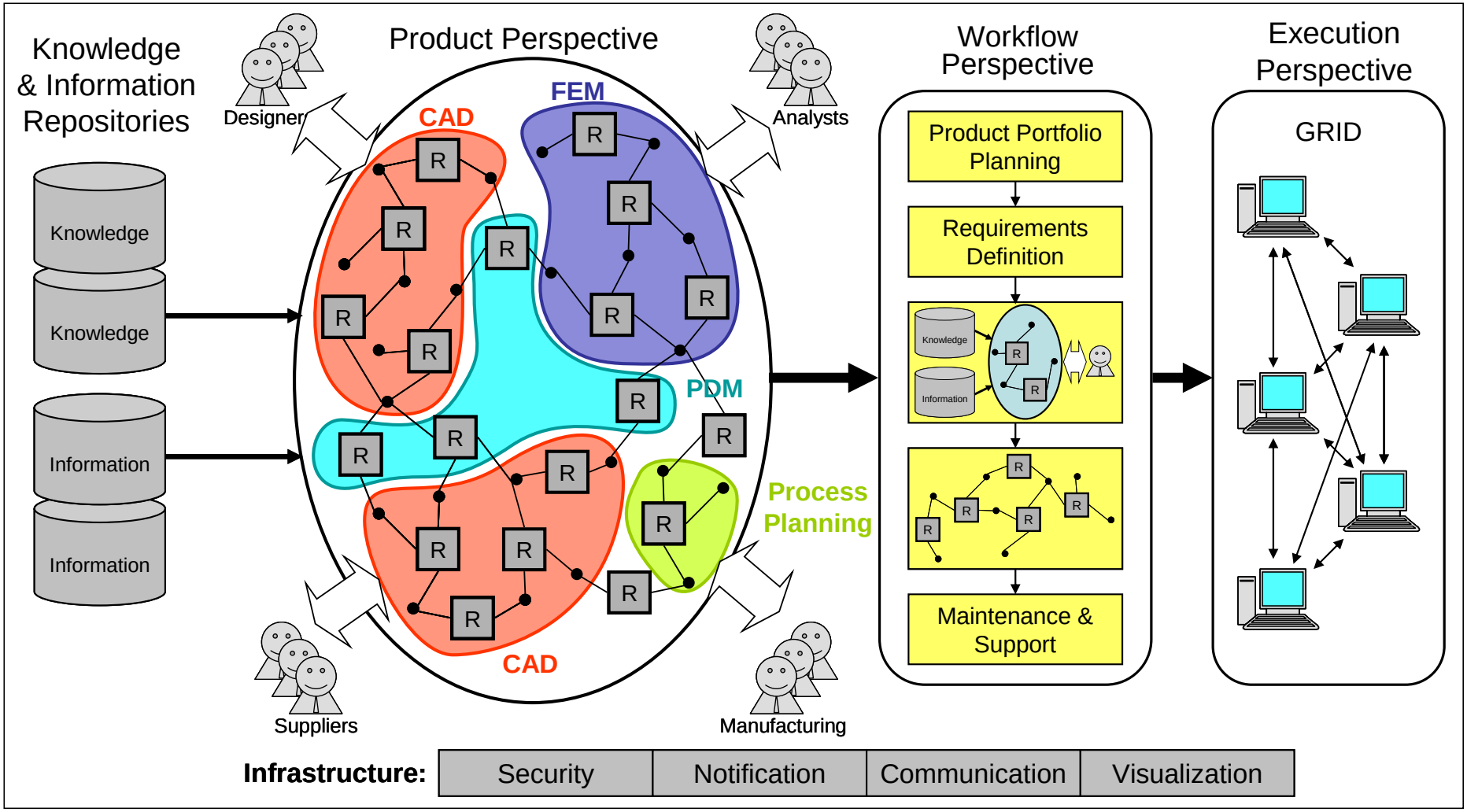
© 2008, Chris Paredis

A Workflow Perspective



© 2008, Chris Paredis

Knowledge & Information in Systems Engineering

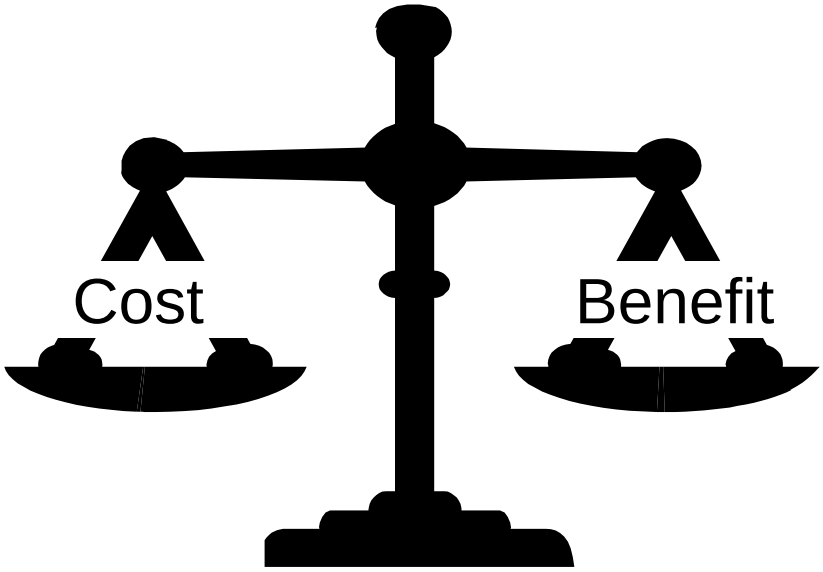


© 2008, Chris Paredis

Information Economics

Information is only valuable
to the extent that it leads to better decisions

- Representation
- Modeling
- Computation



Better design decision
→ higher utility

Design Research: Models, Methods, Tools
for Shifting the Balance towards Higher Value

© 2008, Chris Paredis

Presentation Overview

- What is Model-Based Systems Engineering? Why MBSE?
- A simple model of knowledge and information in MBSE
- Which kinds of knowledge and information?
  - Information schema
  - Language mappings
  - Abstractions and idealizations
  - Analysis patterns
  - Synthesis patterns
  - Design Workflow
- Challenges and Summary

© 2008, Chris Paredis

Information Schema

- Traditionally: tool-specific proprietary schema
  - CAD
  - Finite element modeling
  - Discrete event simulation
  - Few formal representations in SE

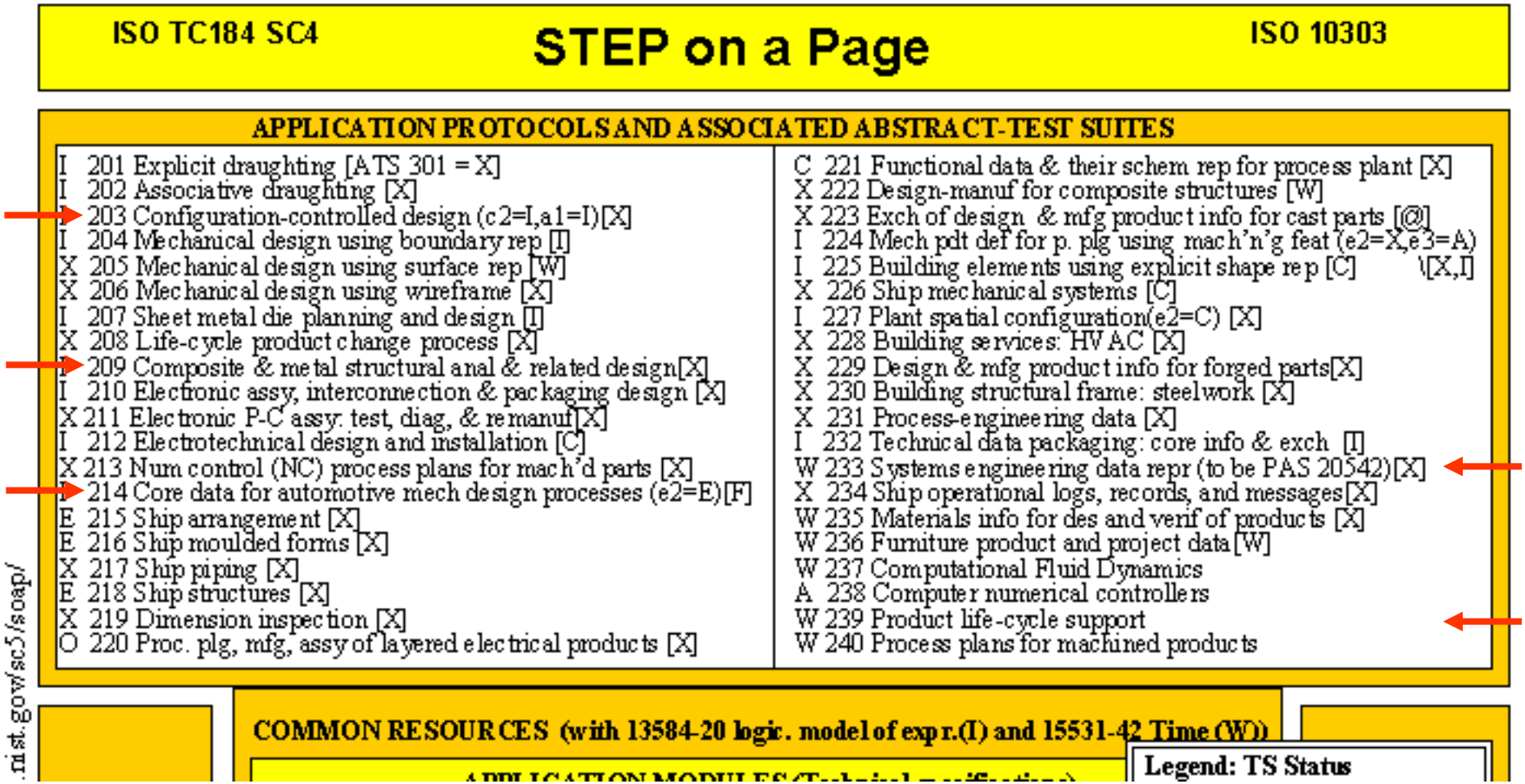
Islands of Knowledge & Information

- Standardization
  - e.g.: STEP (ISO 10303) – Standard for the Exchange of Product data
  - Intent: Reduce the number of translators from N^2 to N
  - Always lagging behind
  - Overlap and inconsistencies

Not all types of information have been modeled formally.
The models are changing over time.

© 2008, Chris Paredis

Example of Standardization: STEP



ISO TC184 SC4

STEP on a Page

ISO 10303

APPLICATION PROTOCOLS AND ASSOCIATED ABSTRACT-TEST SUITES

I 201 Explicit draughting [ATS 301 = X]
I 202 Associative draughting [X]
I 203 Configuration-controlled design (c2=I,a1=I)[X]
I 204 Mechanical design using boundary rep [I]
X 205 Mechanical design using surface rep [W]
X 206 Mechanical design using wireframe [X]
I 207 Sheet metal die planning and design [I]
X 208 Life-cycle product change process [X]
I 209 Composite & metal structural anal & related design[X]
I 210 Electronic assy, interconnection & packaging design [X]
X 211 Electronic P-C assy: test, diag, & remanuf[X]
I 212 Electrotechnical design and installation [C]
X 213 Num control (NC) process plans for mach'd parts [X]
I 214 Core data for automotive mech design processes (e2=E)[F]
E 215 Ship arrangement [X]
E 216 Ship moulded forms [X]
X 217 Ship piping [X]
E 218 Ship structures [X]
X 219 Dimension inspection [X]
O 220 Proc. plg, mfg, assy of layered electrical products [X]
C 221 Functional data & their schem rep for process plant [X]
X 222 Design-manuf for composite structures [W]
X 223 Exch of design & mfg product info for cast parts [@]
I 224 Mech pdt def for p. plg using mach'n'g feat (e2=X,e3=A)
I 225 Building elements using explicit shape rep [C] \[X,I]
X 226 Ship mechanical systems [C]
I 227 Plant spatial configuration(e2=C) [X]
X 228 Building services: HVAC [X]
X 229 Design & mfg product info for forged parts[X]
X 230 Building structural frame: steelwork [X]
X 231 Process-engineering data [X]
I 232 Technical data packaging: core info & exch [I]
W 233 Systems engineering data repr (to be PAS 20542)[X]
X 234 Ship operational logs, records, and messages[X]
W 235 Materials info for des and verif of products [X]
W 236 Furniture product and project data[W]
W 237 Computational Fluid Dynamics
A 238 Computer numerical controllers
W 239 Product life-cycle support
W 240 Process plans for machined products

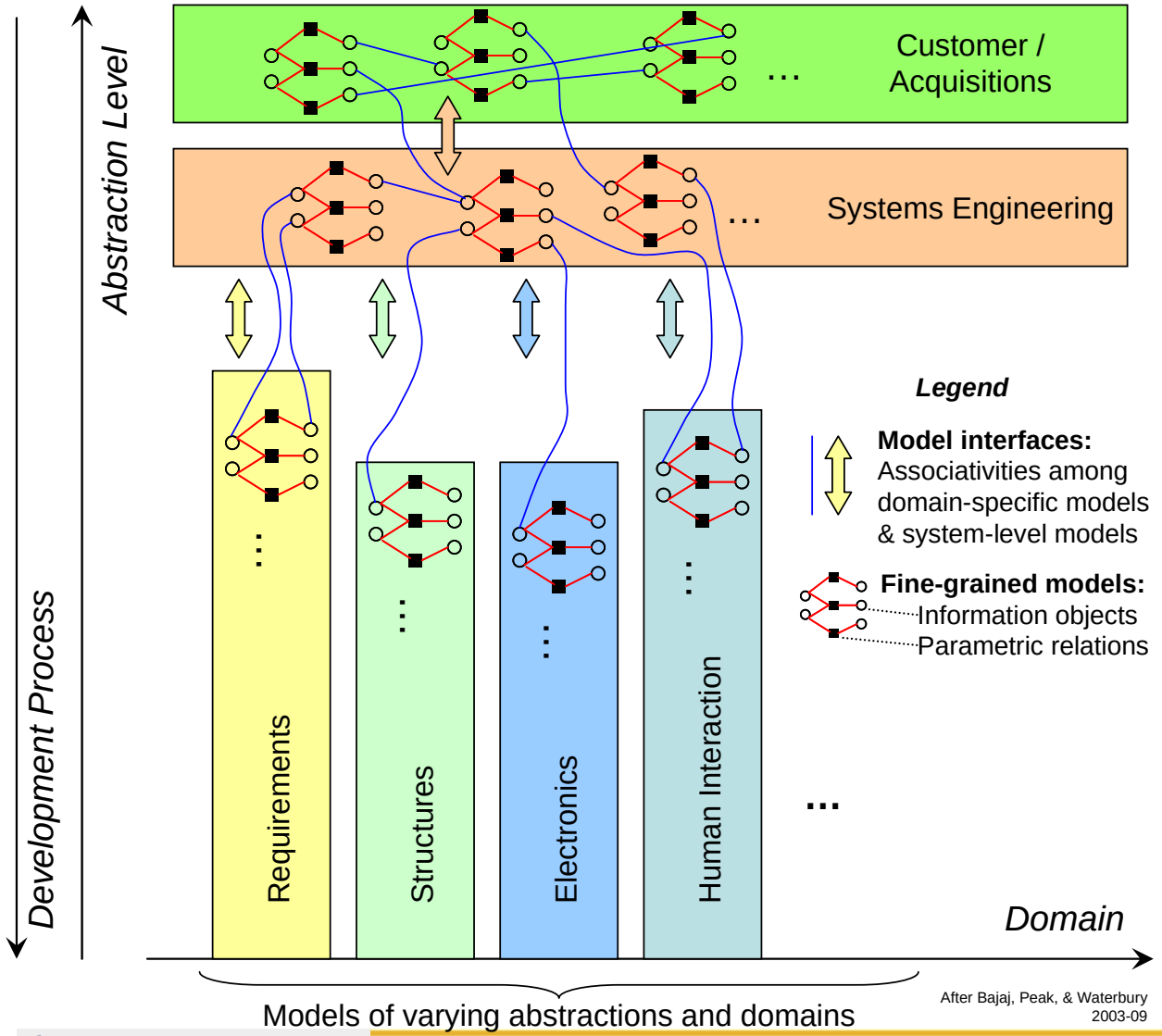
.nist.gov/sc5/soap/

COMMON RESOURCES (with 13584-20 logic. model of expr.(I) and 15531-42 Time (W))

Legend: TS Status

© 2008, Chris Paredis

Federated Model (Russell Peak et al.)

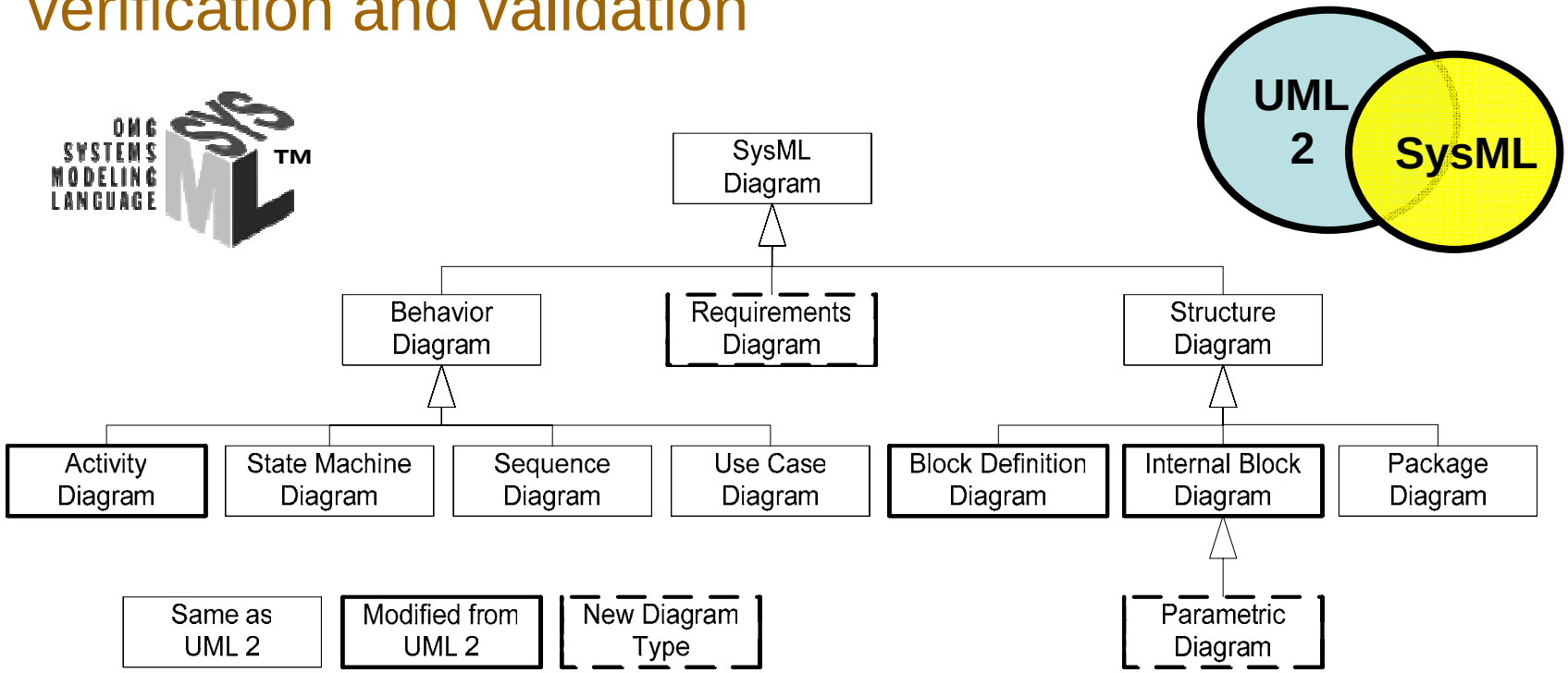


- One single master model is impossible / impractical
  - Evolution
  - Different levels of abstraction
- Relationships must be *fine-grained*
  - Not file-based

© 2008, Chris Paredis

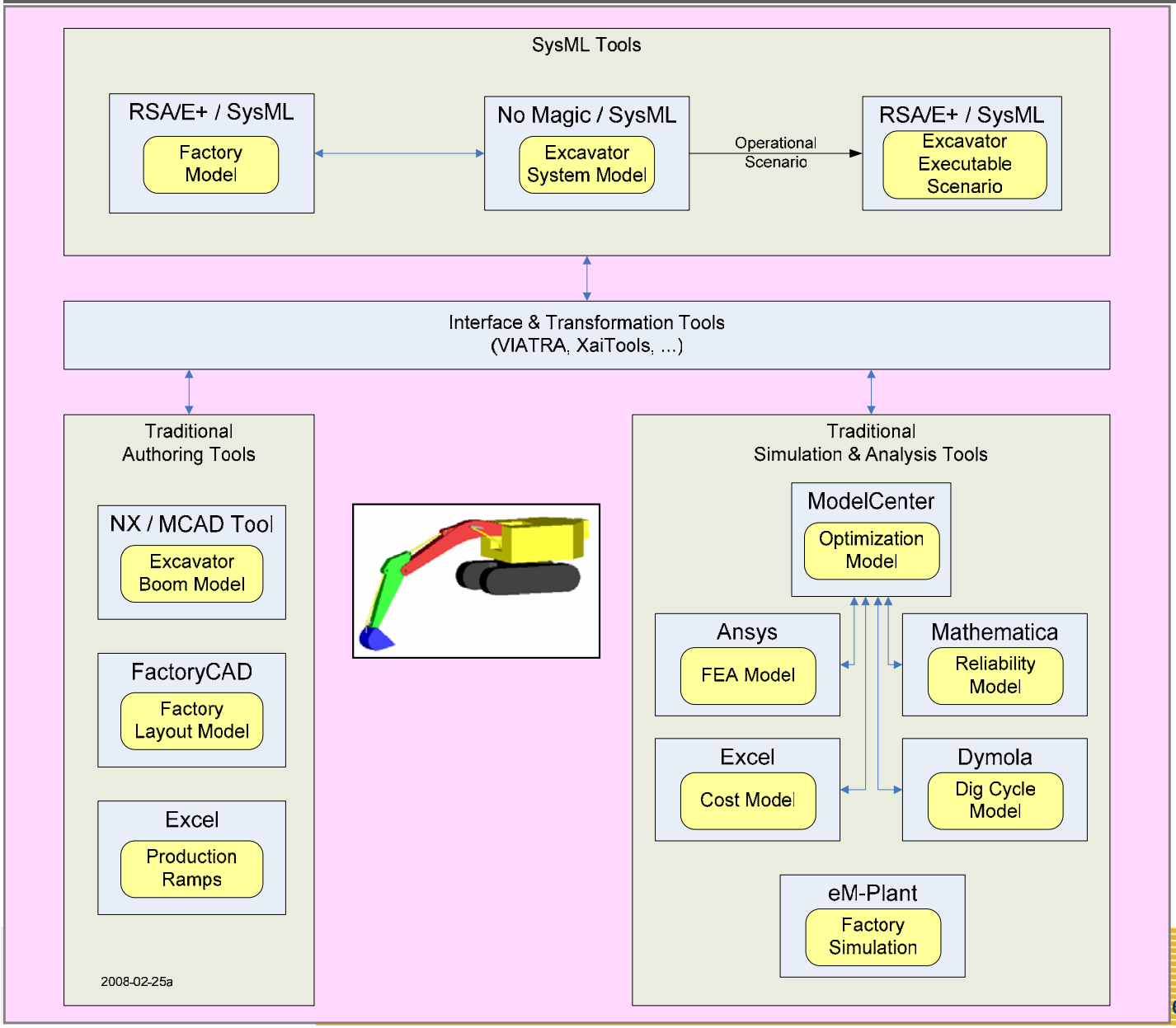
SysML as a Federated Model for MBSE

- Visual information modeling for systems engineering
- Based on UML 2
- Used to support system specification, analysis, design, verification and validation



© 2008, Chris Paredis

GIT Testbed: Tools and Models



- Authoring in native tools
- Integration in SysML
  - Different levels of abstraction
- Language Mapping
  - Graph transformations (e.g., Modelica)
  - Direct interface (e.g., Excel)

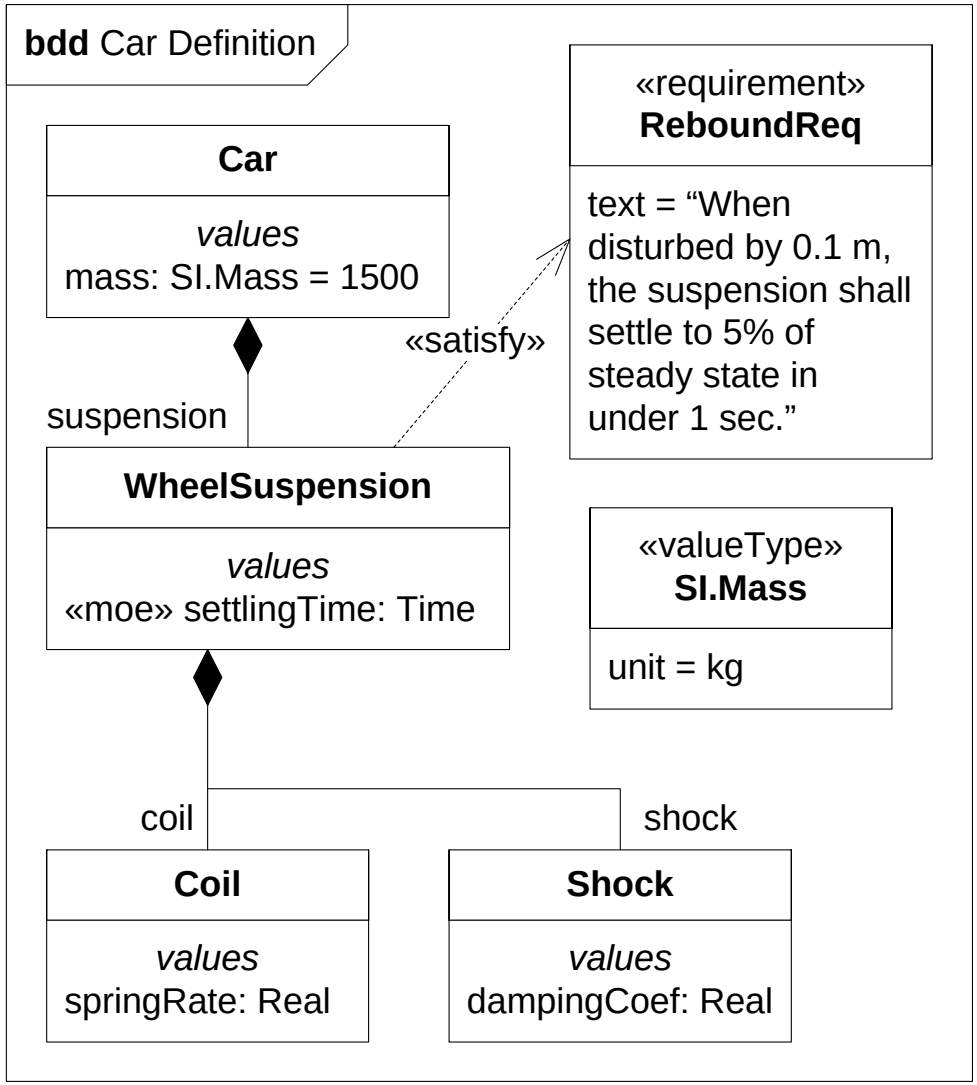
Related Work: Model Integration with SysML

- "…SysML is intended to unify the diverse modeling languages used by systems engineers." (SysML Specification, 2007)
- Constrained Objects (Peak et al., 2001, 2005)
- ModelicaML (Pop et al., 2007)
- UML^{H} (Nytsch-Geusen, 2007)

© 2008, Chris Paredis

Relevant SysML Modeling Constructs

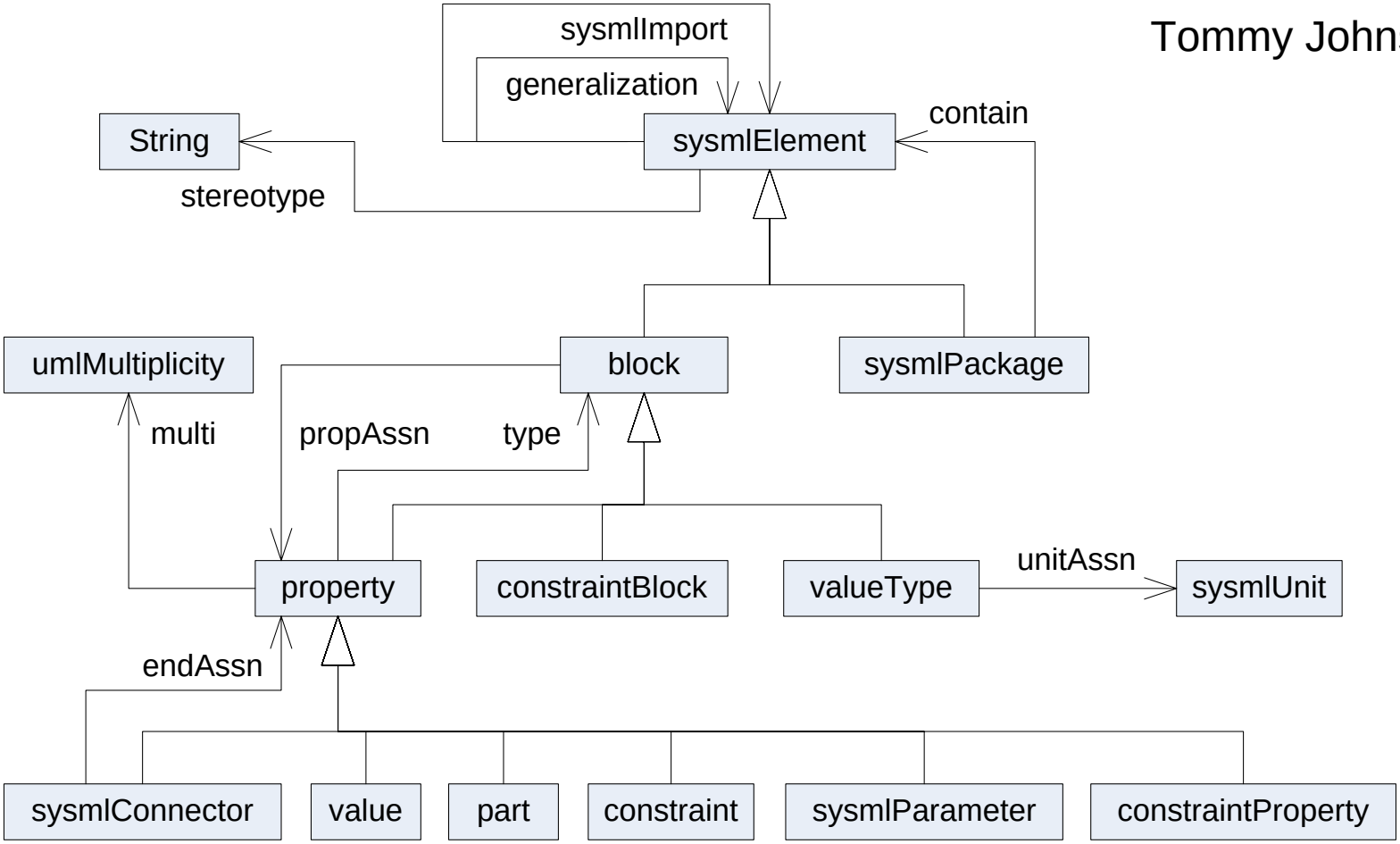
- Blocks
- Value types
- Part properties
- Value properties
- Stereotypes



© 2008, Chris Paredis

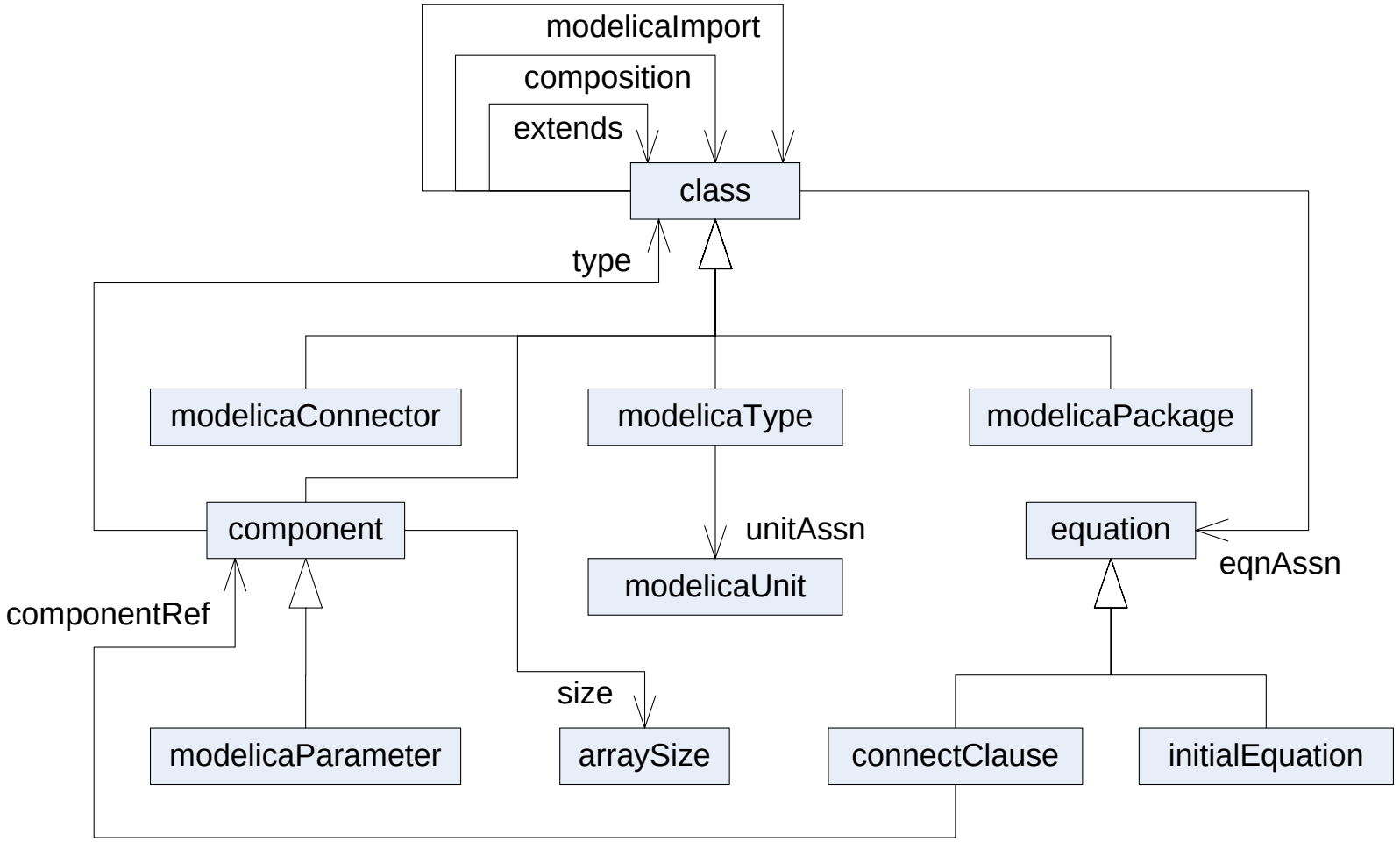
The SysML Metamodel

MS work by
Tommy Johnson



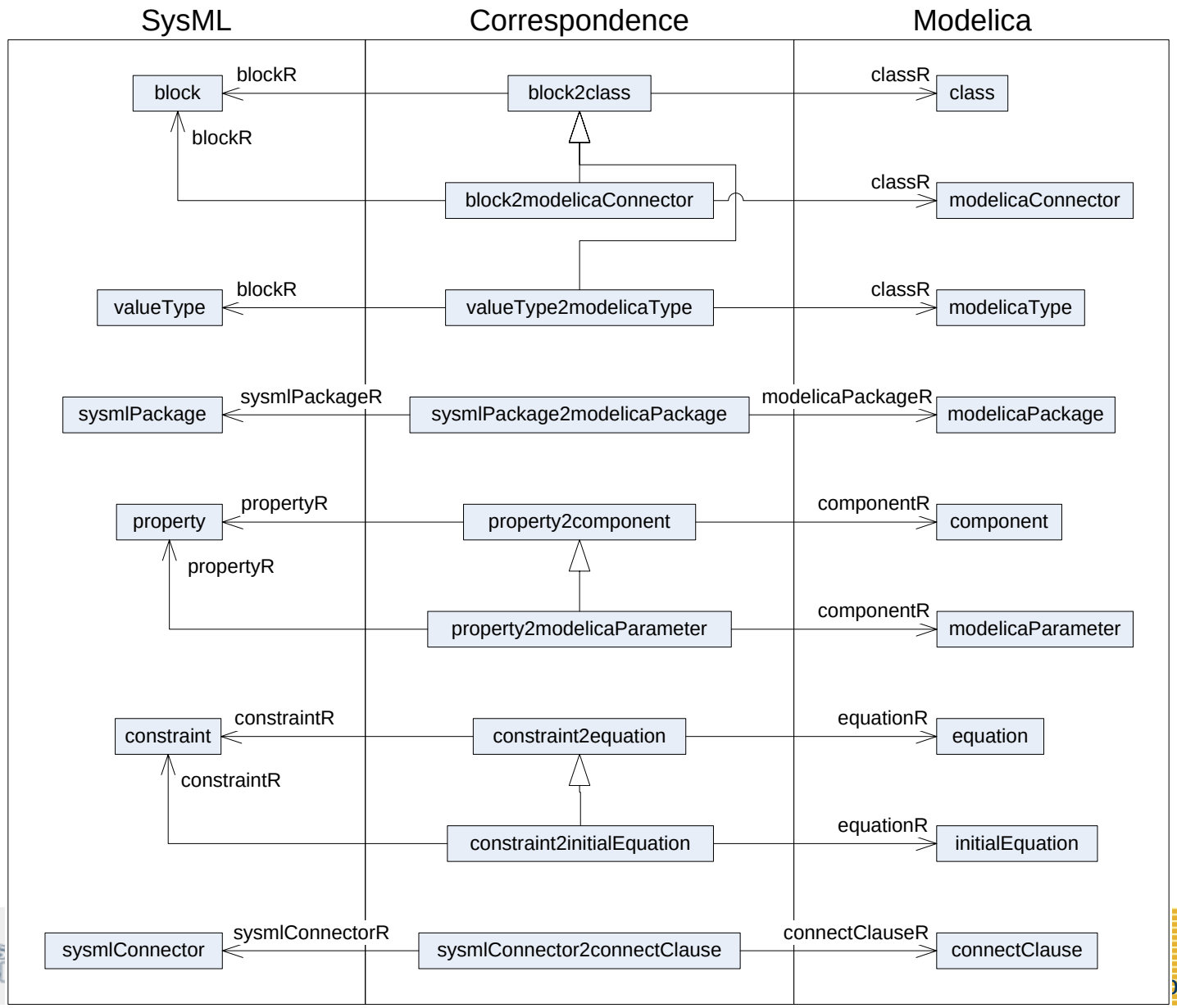
© 2008, Chris Paredis

The Modelica Metamodel



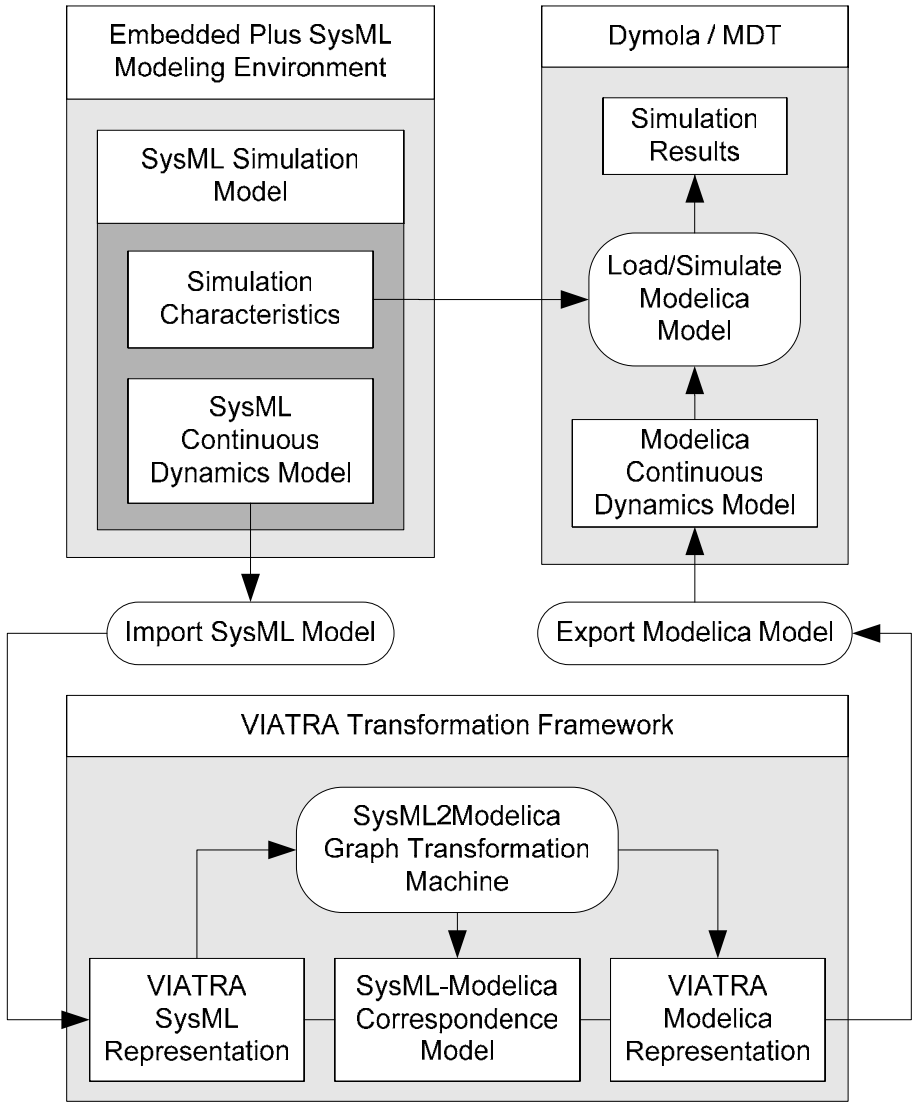
© 2008, Chris Paredis

The Correspondence Metamodel



Implementation in RSD/Eclipse

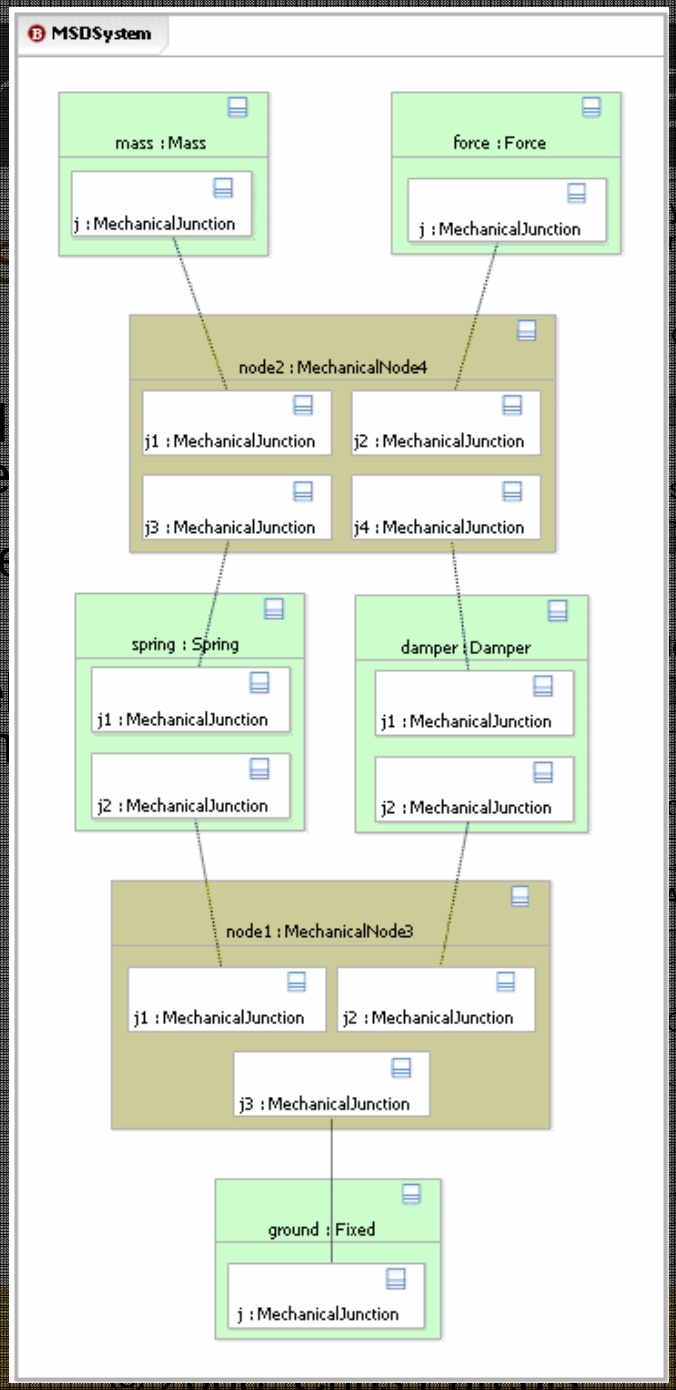
- Created Java software that …
  - Queries Embedded Plus (E+) SysML CD model
  - Transforms model in VIATRA2 framework
  - Generates Modelica code & simulates in Dymola



© 2008, Chris Paredis

Implem Eclipse

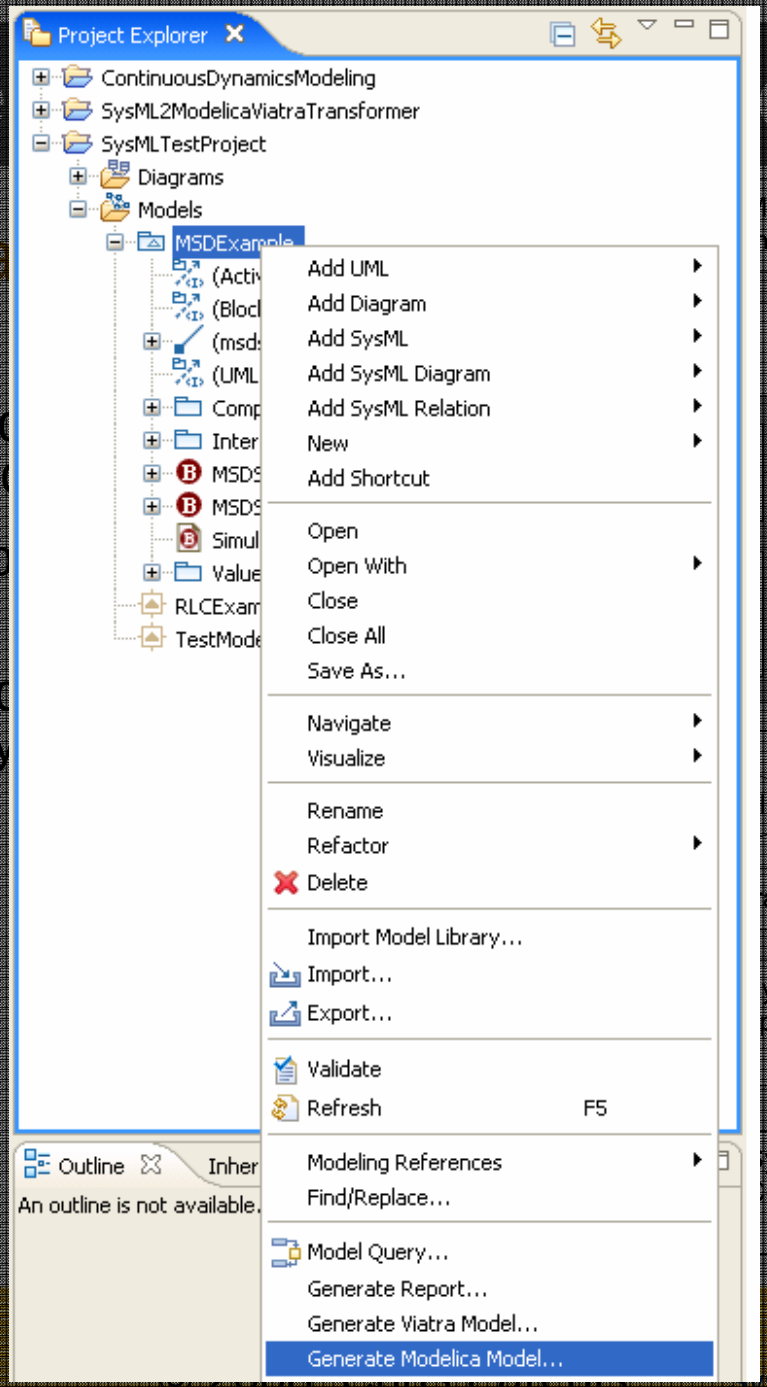
- Created Java s
  that …
  - Queries Embedd
    SysML CD mode
  - Transforms mode
    framework
  - Generates Mode
    simulates in Dym



© 2009, Chris Paredis

Imple…clipse

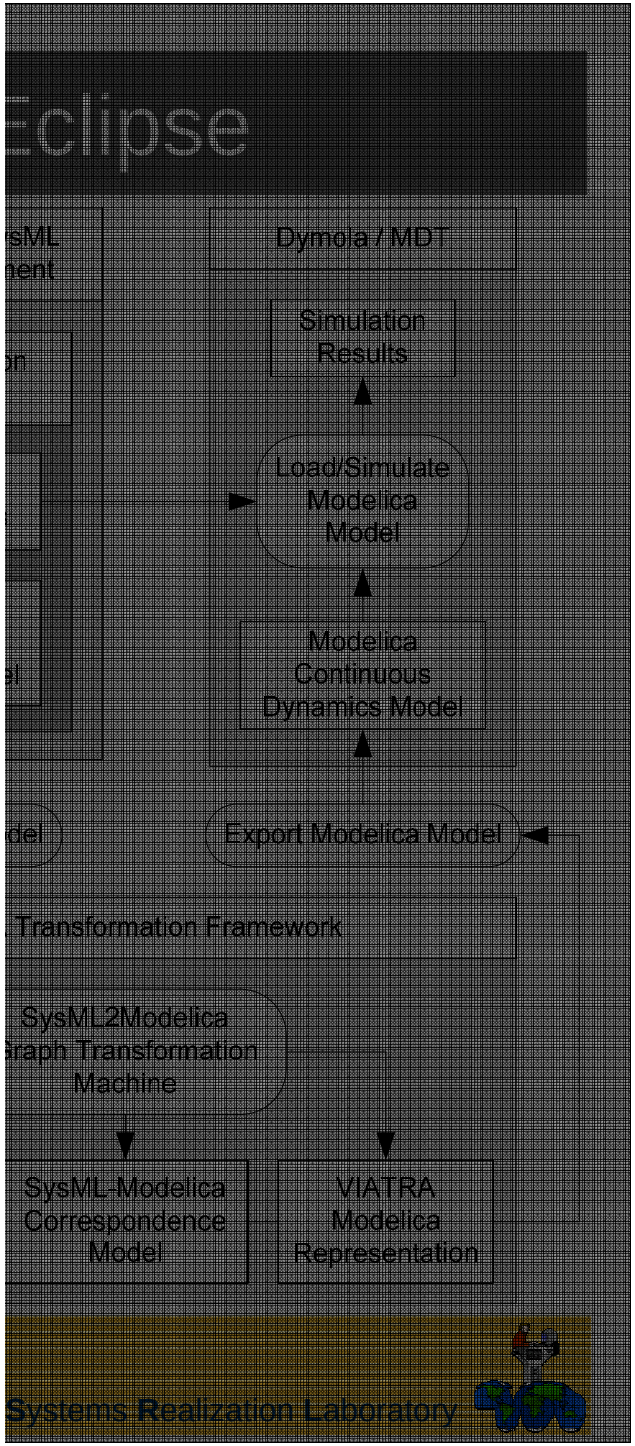
- Created Java… that …
  - Queries Embed… SysML CD mo…
  - Transforms mo… framework
  - Generates Mo… simulates in Dy…



Implem … Eclipse

- Created Java s… that …
  - Queries Embedd… SysML CD mode…
  - Transforms mode… framework
  - Generates Model… simulates in Dym…

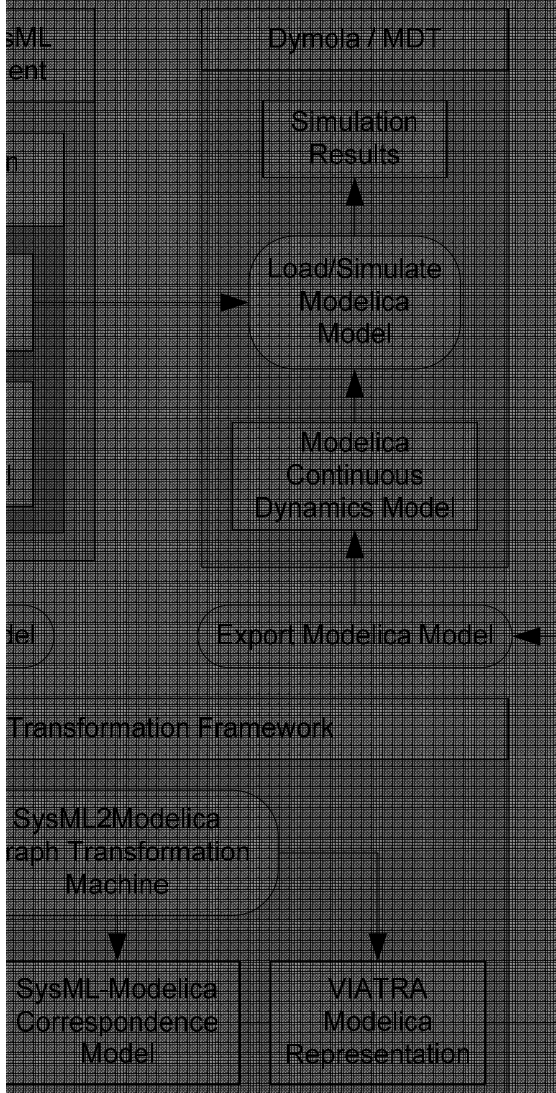
transformSpace : entity
- correspondenceModels : entity
- modelicaModels : entity
- sysmlModels : entity
  - MSDExample : sysmlPackage
    - uN5864_6b (-> MSDDefinition) : contain
    - MSDDefinition : sysmlPackage
      - uN5866_6b (-> ValueTypes) : contain
      - uN5902_6b (-> Components) : contain
      - uN5914_6b (-> Interfaces) : contain
      - uN5922_6b (-> MSDSystem) : contain
      - Components : sysmlPackage
      - Interfaces : sysmlPackage
      - MSDSystem : block
        - uN6056_6b (-> mass) : propAssn
        - uN6059_6b (-> spring) : propAssn
        - uN6062_6b (-> damper) : propAssn
        - uN6065_6b (-> ground) : propAssn
        - uN6068_6b (-> force) : propAssn
        - uN6071_6b (-> node1) : propAssn
        - uN6074_6b (-> node2) : propAssn
        - uN6077_6b (-> massParameter) : propAssn
        - uN6323_6b (-> Connector1) : propAssn
        - uN6327_6b (-> Connector2) : propAssn
        - uN6331_6b (-> Connector3) : propAssn
        - uN6335_6b (-> Connector4) : propAssn
        - uN6339_6b (-> Connector5) : propAssn
        - uN6343_6b (-> Connector6) : propAssn
        - uN6347_6b (-> Connector7) : propAssn
        - Connector1 : sysmlConnector
        - Connector2 : sysmlConnector
        - Connector3 : sysmlConnector
        - Connector4 : sysmlConnector
        - Connector5 : sysmlConnector
        - Connector6 : sysmlConnector
        - Connector7 : sysmlConnector
        - damper : part
        - force : part
        - ground : part
        - mass {(m = massParameter)} : part
        - massParameter {10} : value
        - node1 : part
        - node2 : part
        - spring : part
      - ValueTypes : sysmlPackage



Implen … Eclipse

- Created Java … that …
  - Queries Embedd… SysML CD mode…
  - Transforms mod… framework
  - Generates Mode… simulates in Dym…

- transformSpace : entity
  - correspondenceModels : entity
  - modelicaModels : entity
    - libraryModels : entity
    - models : entity
      - MSDExample : modelicaPackage
        - uN5864_6b (-> MSDDefinition) : composition
        - MSDDefinition : modelicaPackage
          - uN5866_6b (-> ValueTypes) : composition
          - uN5902_6b (-> Components) : composition
          - uN5914_6b (-> Interfaces) : composition
          - uN5922_6b (-> MSDSystem) : composition
          - Components : modelicaPackage
          - Interfaces : modelicaPackage
          - MSDSystem : class
            - uN6056_6b (-> mass) : composition
            - uN6059_6b (-> spring) : composition
            - uN6062_6b (-> damper) : composition
            - uN6065_6b (-> ground) : composition
            - uN6068_6b (-> force) : composition
            - uN6071_6b (-> node1) : composition
            - uN6074_6b (-> node2) : composition
            - uN6077_6b (-> massParameter) : composition
            - uN6323_6b (-> Connector1) : eqnAssn
            - uN6327_6b (-> Connector2) : eqnAssn
            - uN6331_6b (-> Connector3) : eqnAssn
            - uN6335_6b (-> Connector4) : eqnAssn
            - uN6339_6b (-> Connector5) : eqnAssn
            - uN6343_6b (-> Connector6) : eqnAssn
            - uN6347_6b (-> Connector7) : eqnAssn
            - Connector1 : connectClause
            - Connector2 : connectClause
            - Connector3 : connectClause
            - Connector4 : connectClause
            - Connector5 : connectClause
            - Connector6 : connectClause
            - Connector7 : connectClause
            - damper : component
            - force : component
            - ground : component
            - mass {(m = massParameter)} : component
            - massParameter {10} : modelicaParameter
            - node1 : component
            - node2 : component
            - spring : component
          - ValueTypes : modelicaPackage
  - sysmlModels : entity



MSDExample.emx | *MSDExample.mo

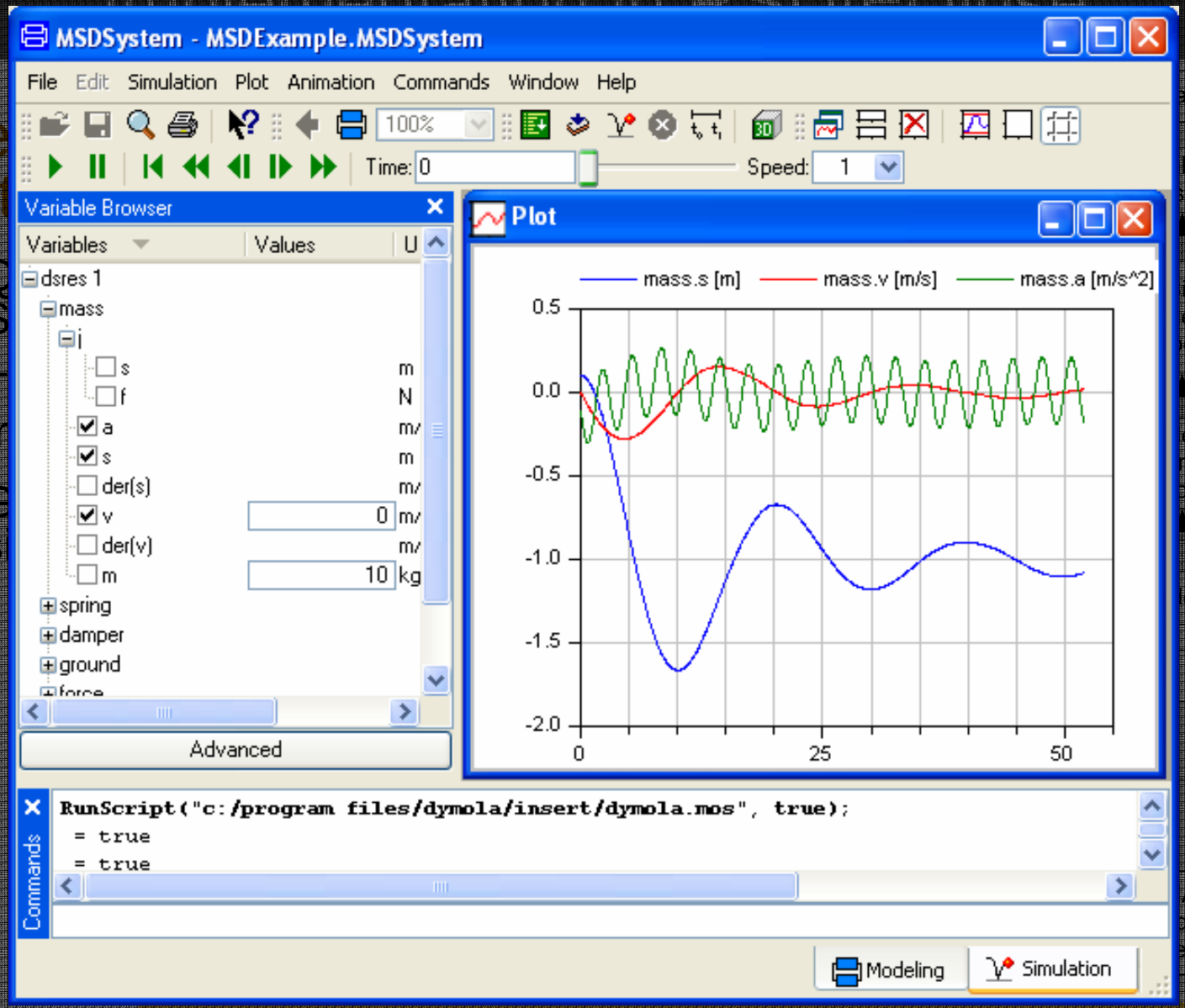
```
package MSDExample
  ...
  package MSDDefinition
    ...
    class MSDSystem
      MSDExample.MSDDefinition.Components.Mass mass(m = massParameter);
      MSDExample.MSDDefinition.Components.Spring spring;
      MSDExample.MSDDefinition.Components.Damper damper;
      MSDExample.MSDDefinition.Components.Fixed ground;
      MSDExample.MSDDefinition.Components.Force force;
      MSDExample.MSDDefinition.Interfaces.MechanicalNode node1;
      MSDExample.MSDDefinition.Interfaces.MechanicalNode4 node2;
      MSDExample.MSDDefinition.ValueTypes.Mass massParameter;
    equation
      connect(ground.j, node1.j3);
      connect(node1.j1, spring.j2);
      connect(damper.j2, node1.j2);
      connect(spring.j1, node2.j3);
      connect(damper.j1, node2.j4);
      connect(node2.j2, force.j);
      connect(node2.j1, mass.j);
    end MSDSystem;

  end MSDDefinition;

end MSDExample;
```


© 2008, Chris Paredis

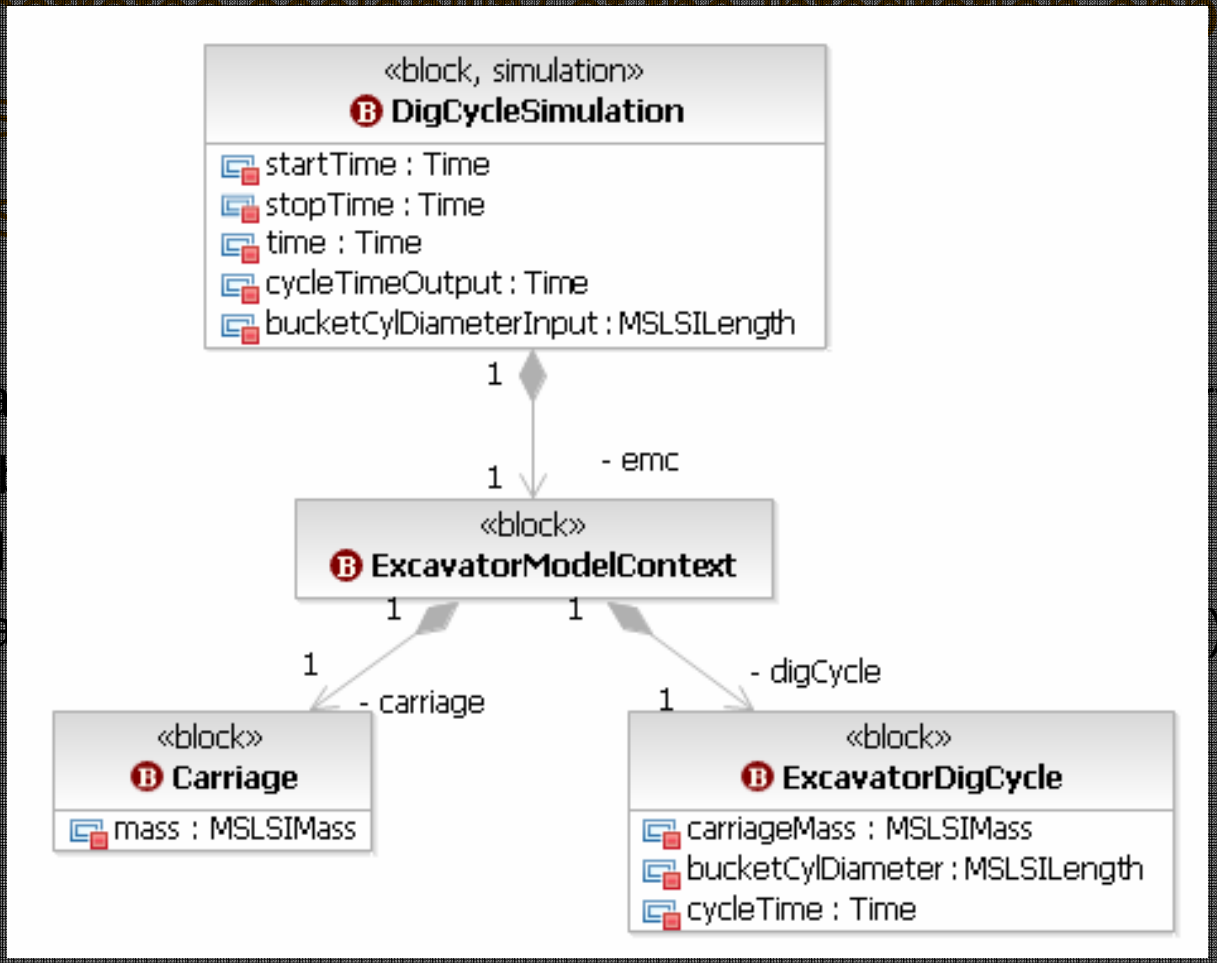
Implementation in DSDEclipse



© 2008, Chris Paredis

The Hydraulically Powered Excavator Example

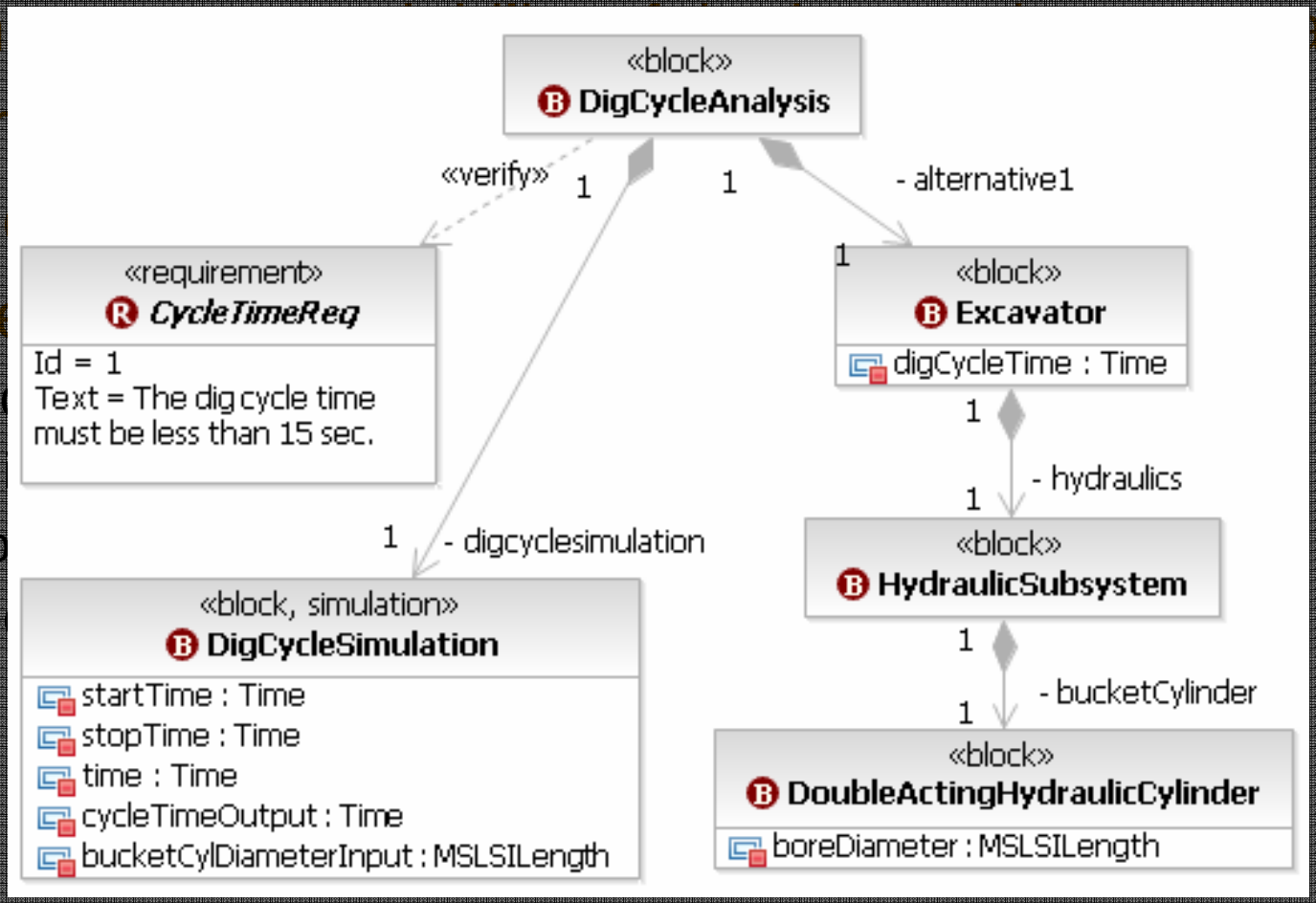
- Demonstrate capability of the integration approach
- Represents
- Depicts
- Model
  - hydra
  - rigid-l
  - world
  - opera



© 2008, Chris Paredis

The Hydraulically Powered Excavator Example

- Dem
- Repr
- Depi
- Mode
  - hy
  - rig
  - wo
  - op



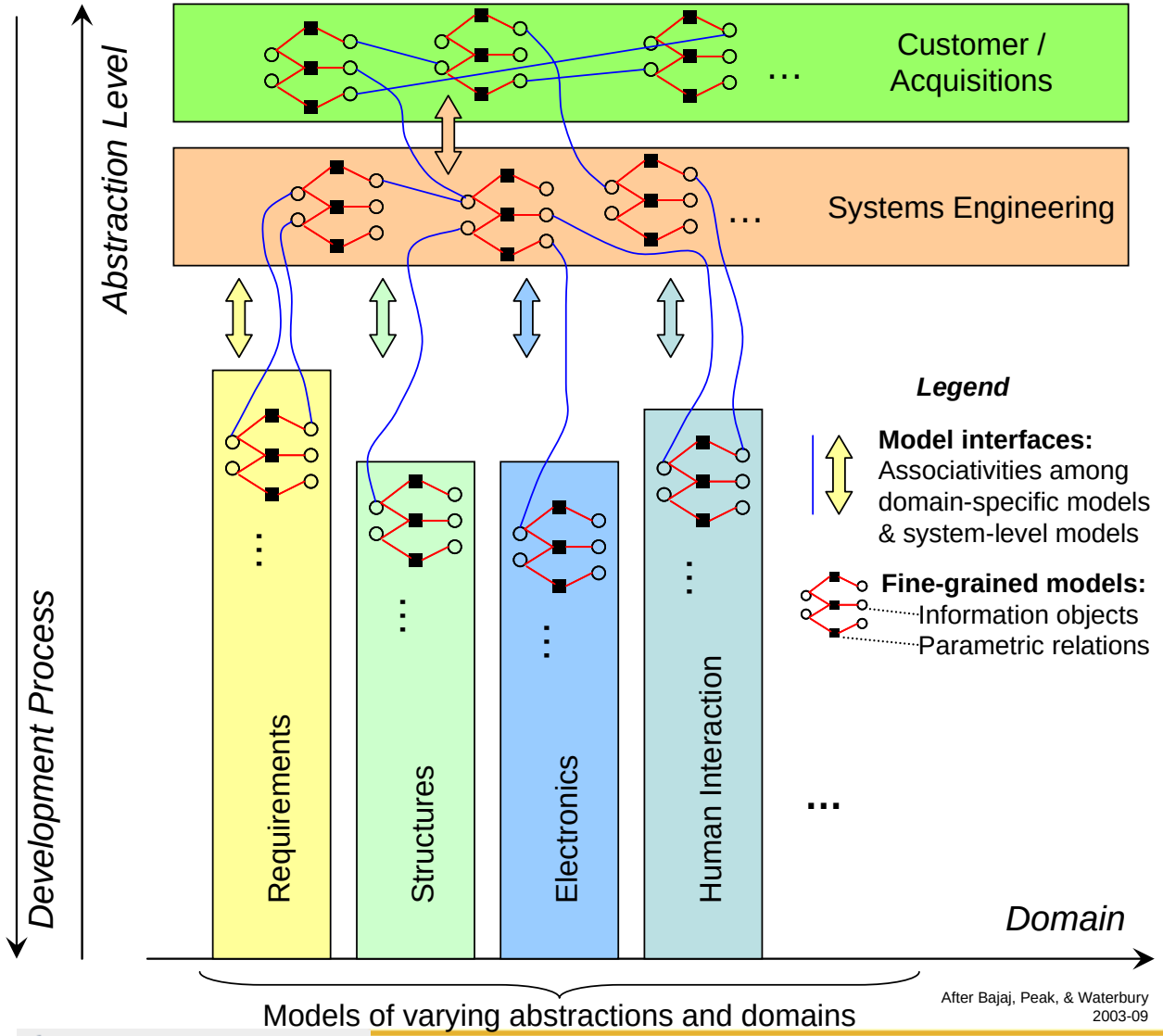
© 2008, Chris Paredis

Presentation Overview

- What is Model-Based Systems Engineering?
  Why MBSE?
- A simple model of knowledge and information in MBSE
- Which kinds of knowledge and information?
  - Information schema
  - Language mappings
  - Abstractions and idealizations
  - Analysis patterns
  - Synthesis patterns
- Managing Design Workflow
- Summary

© 2008, Chris Paredis

Federated Model (Russell Peak et al.)

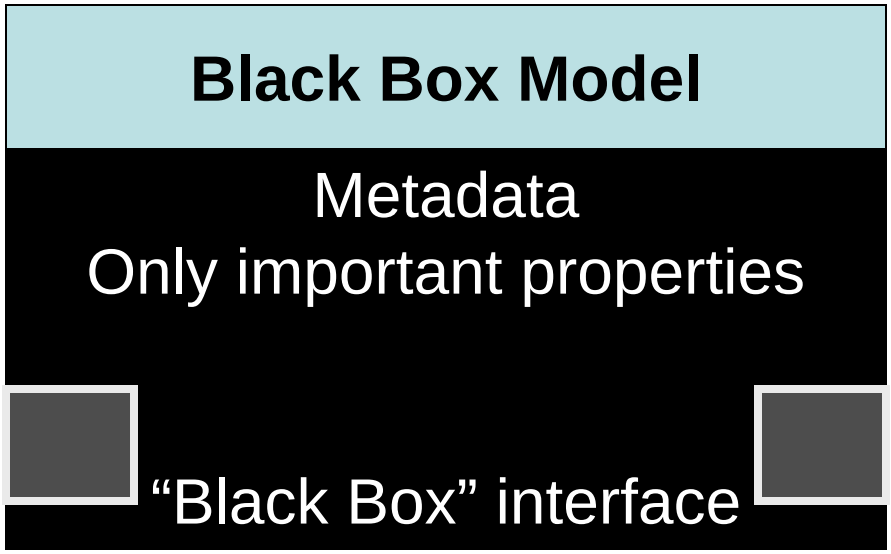


- Decomposability → coupling between detailed models is limited
- Abstraction → hide the details

© 2008, Chris Paredis

Abstraction: Integrating “Black Box” Models

- Represents a pre-existing model using SysML constructs
- Hides details of the model & only exposes important aspects of model
- The «external» stereotype
- Modelica-specific system nodes & the «connectClause» stereotype



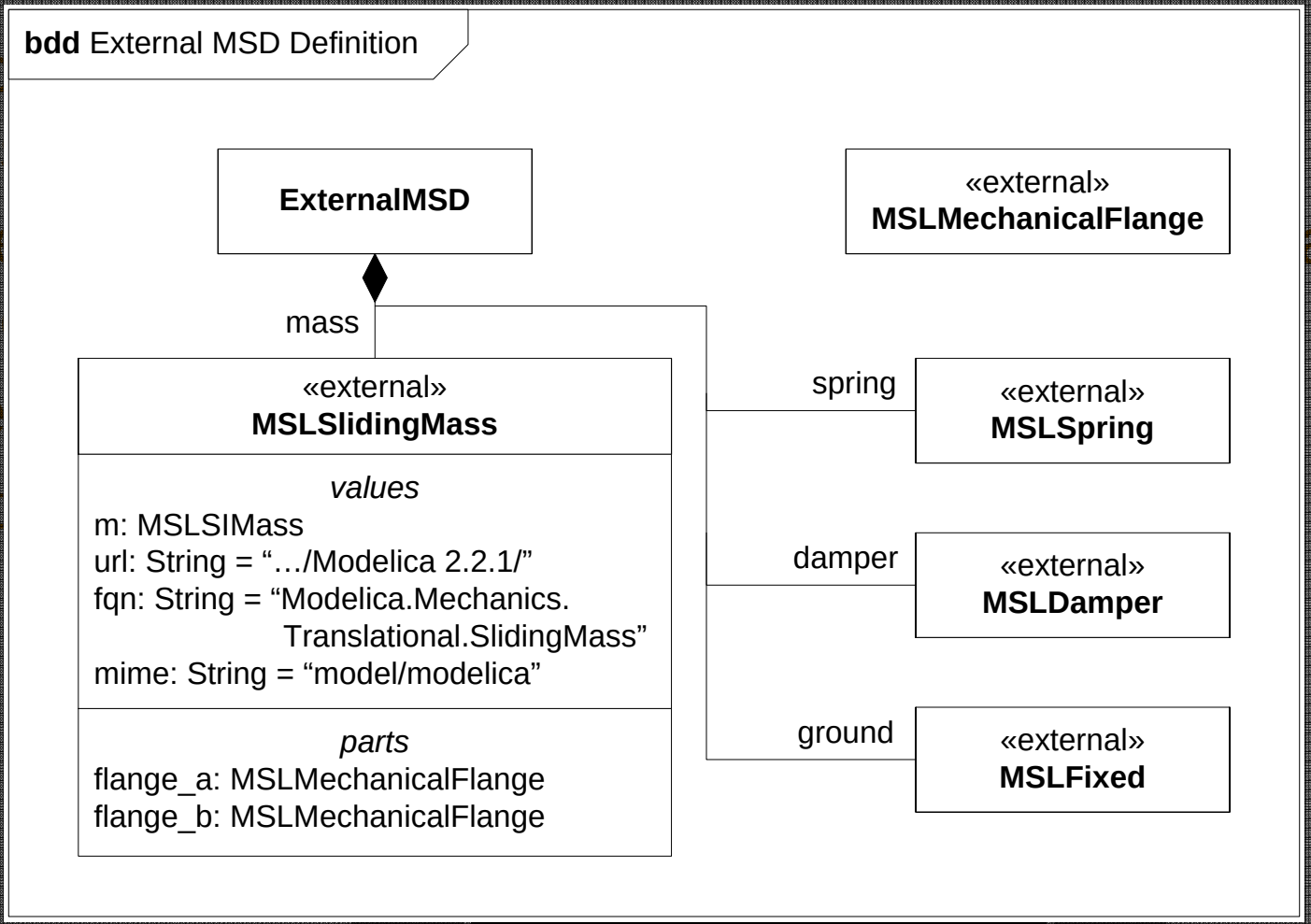
© 2008, Chris Paredis

Integrating "Black Box" Models

- Rep
con
- Hid
asp
The
Mo
«co

ant

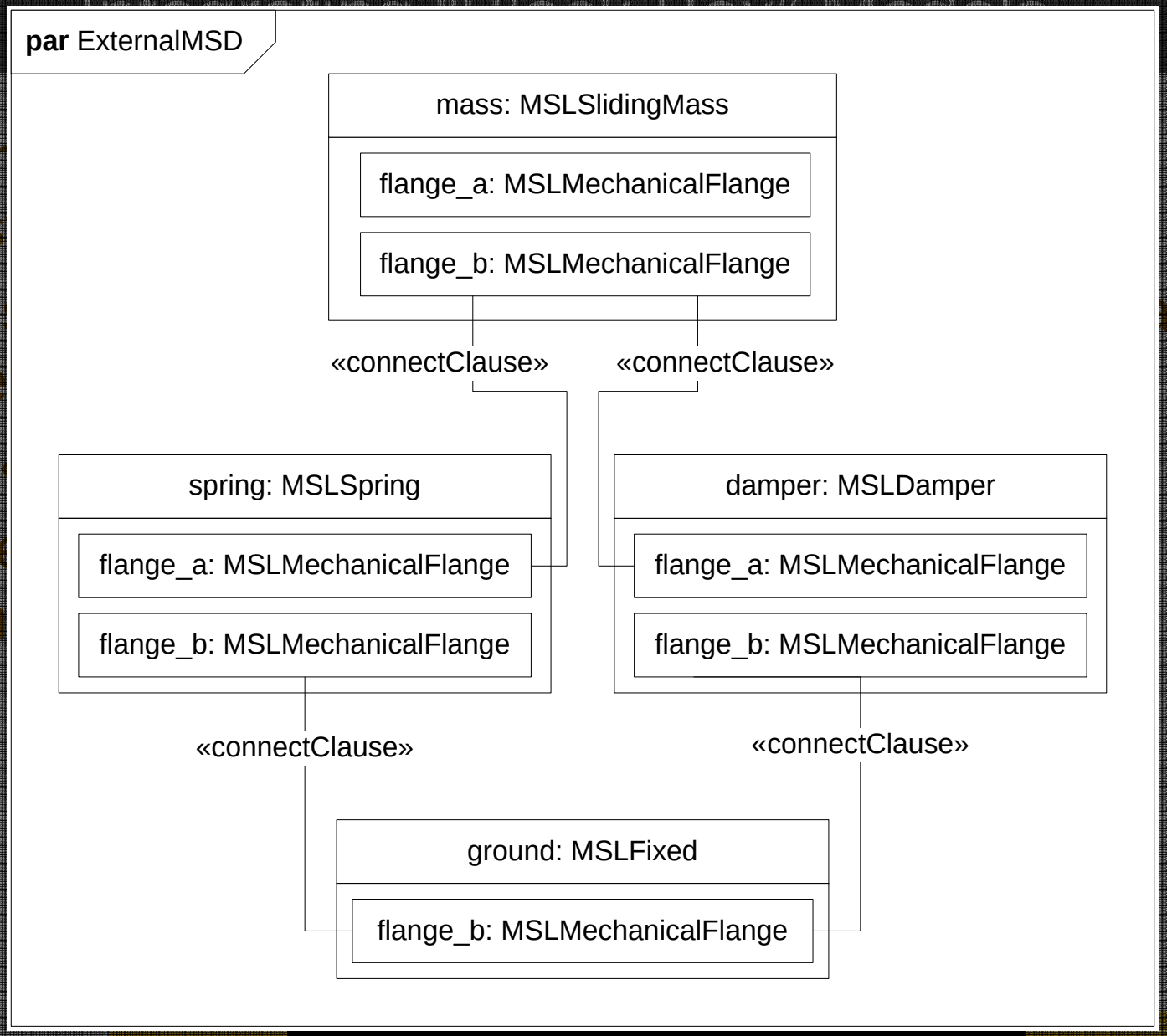
MIA



"Black Box" interface

© 2008, Chris Paredis

- Repr
cons
- Hide
aspe
The
Mod
«con



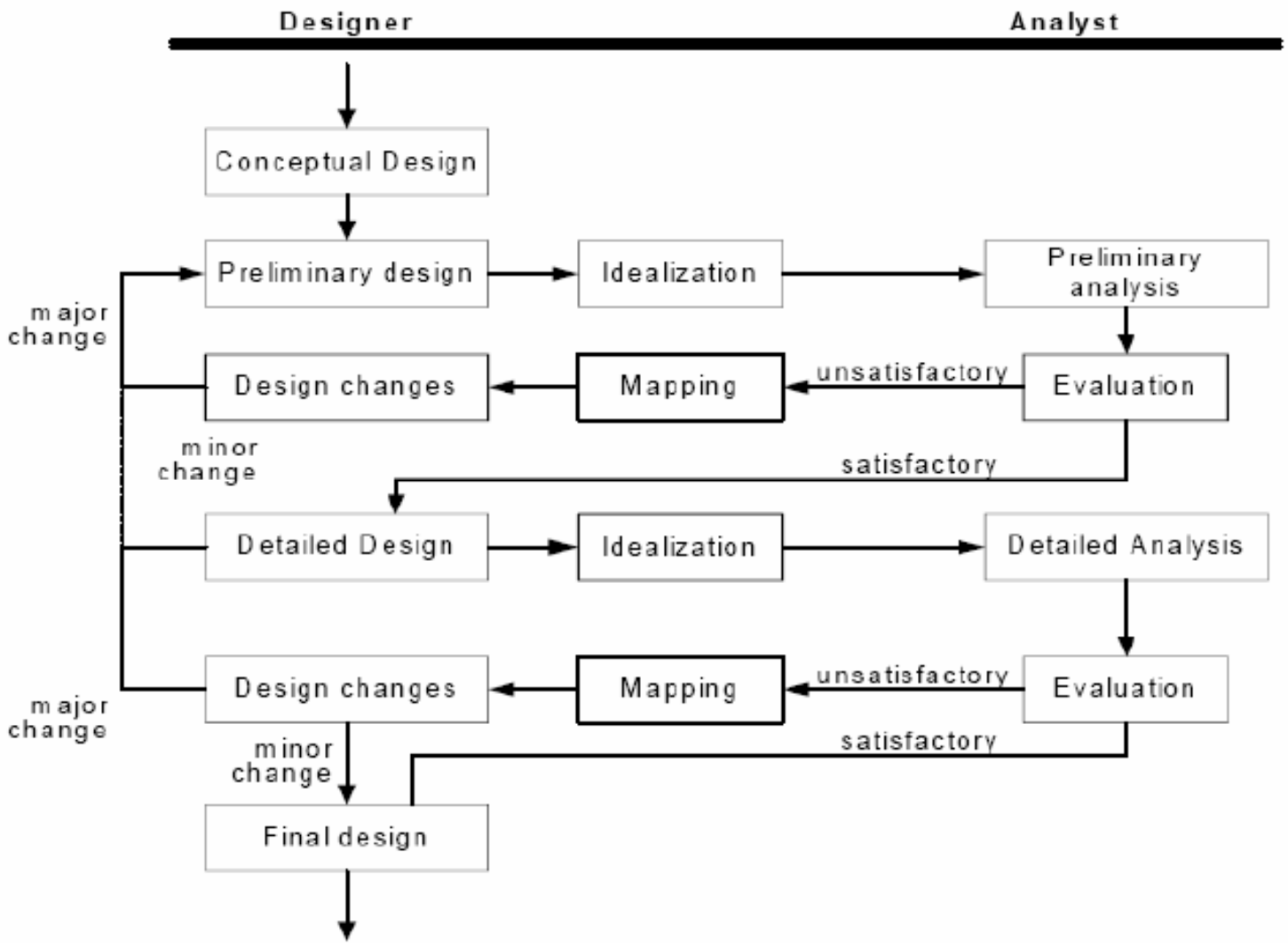
© 2008, Chris Paredis

Presentation Overview

- What is Model-Based Systems Engineering?
  Why MBSE?
- A simple model of knowledge and information in MBSE
- Which kinds of knowledge and information?
  - Information schema
  - Language mappings
  - Abstractions and idealizations
  - Analysis patterns
  - Synthesis patterns
- Managing Design Workflow
- Summary

© 2008, Chris Paredis

Idealizations



Work by Fenves, Choi, et al.

© 2008, Chris Paredis

Idealizations

- Different from just abstraction:
  - Additional knowledge is necessary
  - Appropriate idealization is context dependent

- Most common for finite element modeling: Transform design model into corresponding analysis model
  - Modify geometry
    - Eliminate details
    - Impose symmetry
    - Determine dimensionality
  - Define materials model
  - Model loading conditions
  - Generate mesh

© 2008, Chris Paredis

Methods for Generating Finite Element Models

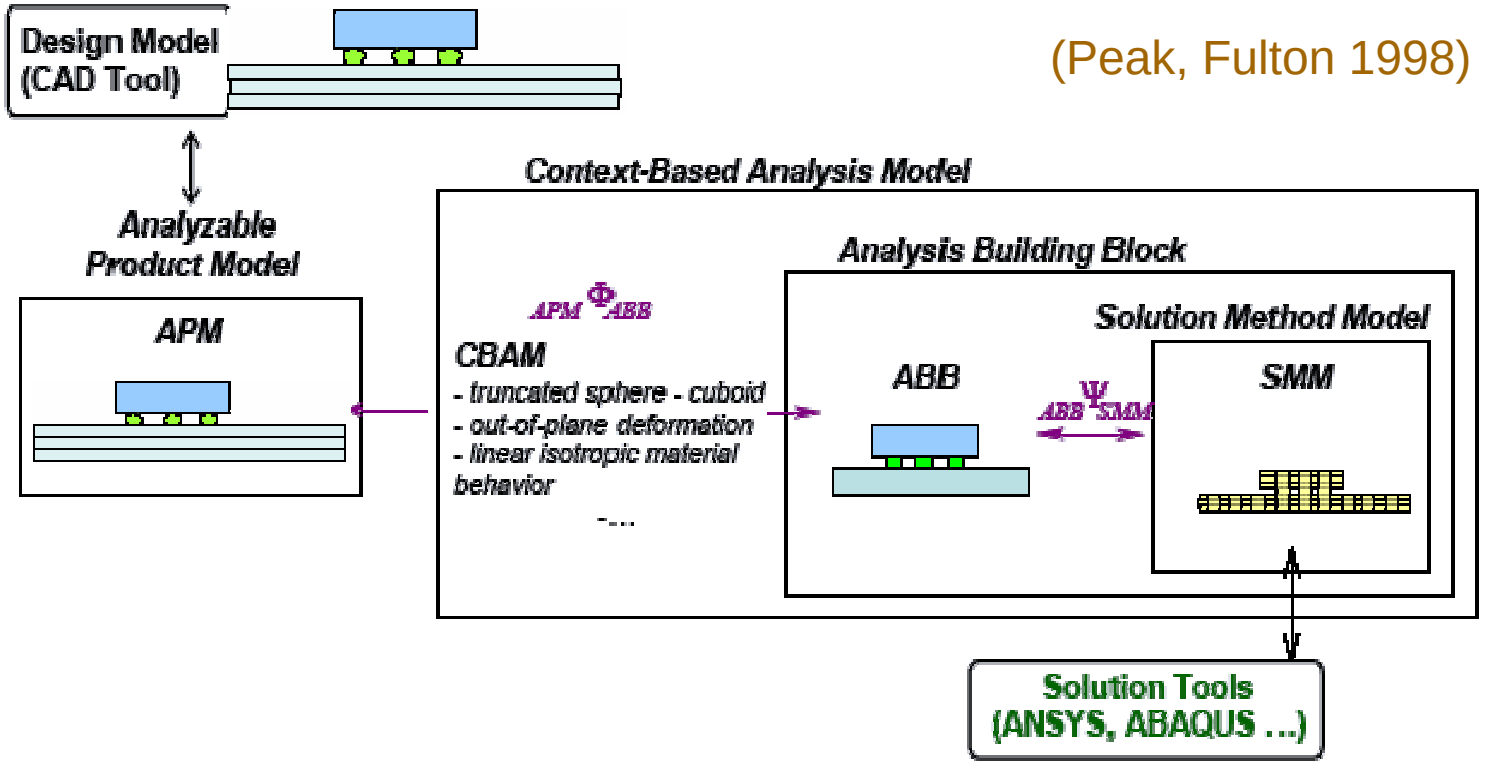
- Direct FEA model creation
  - Import design from CAD tool
  - Modify geometry (as per idealizations)
  - Select elements to mesh (as per idealizations)
  - Mesh

- Script-based FEA model structure creation
  - Author a solver-specific script
    - Idealized geometry, material properties, load, boundary (and/or initial conditions)
    - Element, mesh size, …
  - Run script for different instances

© 2008, Chris Paredis

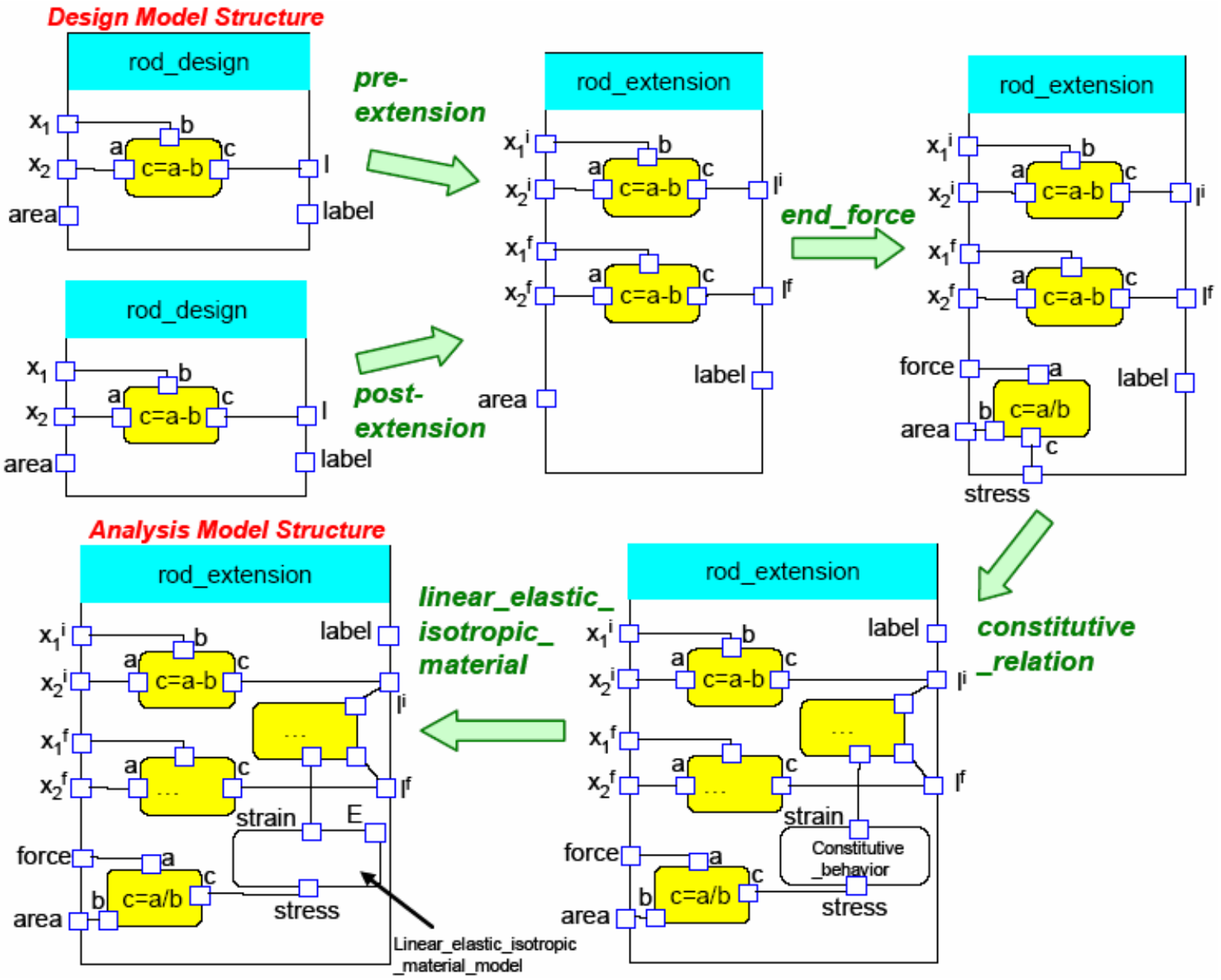
Methods for Generating Finite Element Models

- Multi-representation architecture-based method
  - 4 stepping-stone model structures
  - Models related by parametric constraints (per idealization) → static
  - Analysis model = assembly of reusable building blocks
  - Solution-approach and method specific analysis models



(Peak, Fulton 1998)

Transform Design Model into Analysis Model

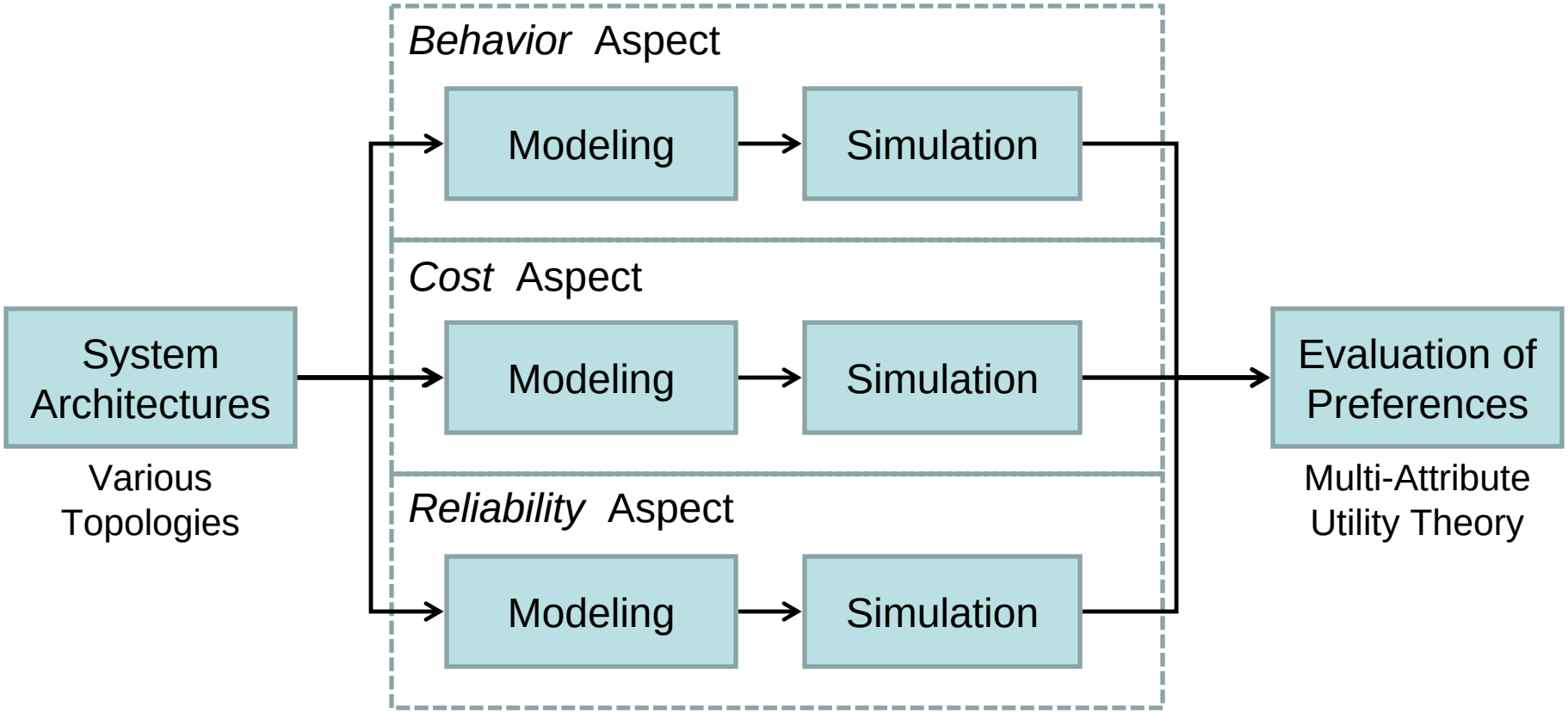


Presentation Overview

- What is Model-Based Systems Engineering?
  Why MBSE?
- A simple model of knowledge and information in MBSE
- Which kinds of knowledge and information?
  - Information schema
  - Language mappings
  - Abstractions and idealizations
  - Analysis patterns
  - Synthesis patterns
  - Design Workflow
- Challenges and Summary

© 2008, Chris Paredis

Scope of the Problem

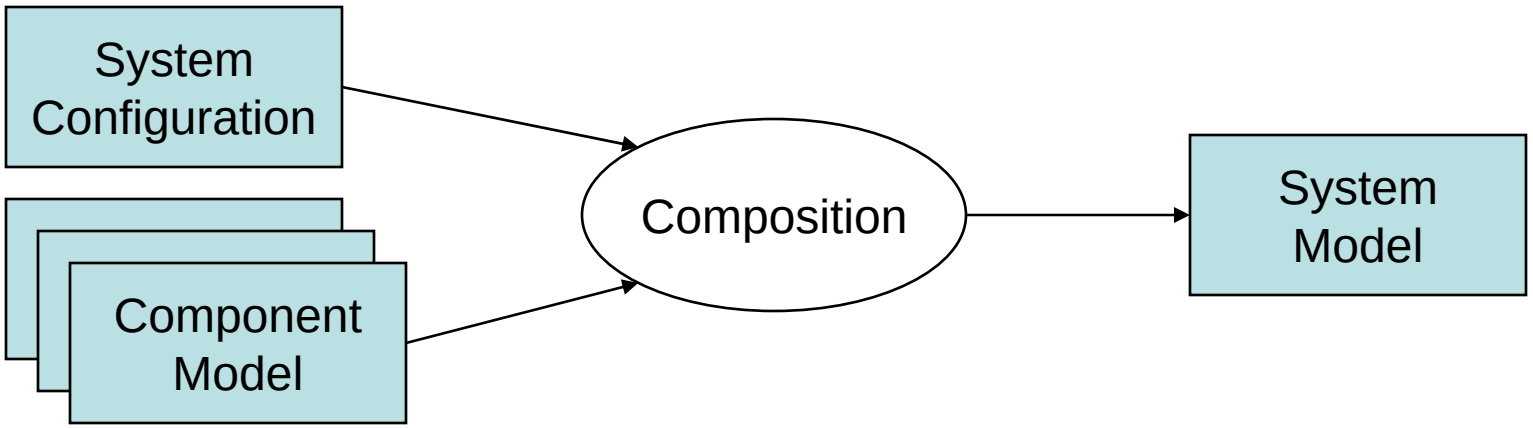


Can we reduce the cost of modeling through model composition?

Analysis Patterns

- System is composition/configuration of subsystems or components
- The subsystems and components are reusable building blocks

- We need to predict the behavior of every configuration considered in design

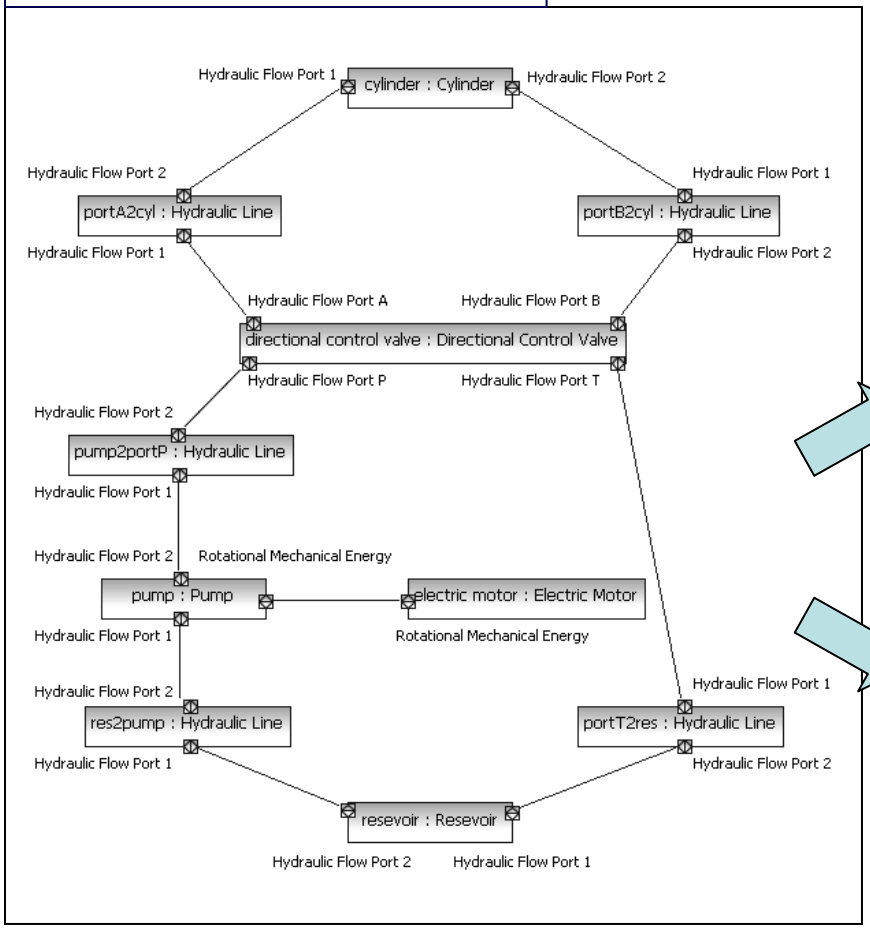


© 2008, Chris Paredis

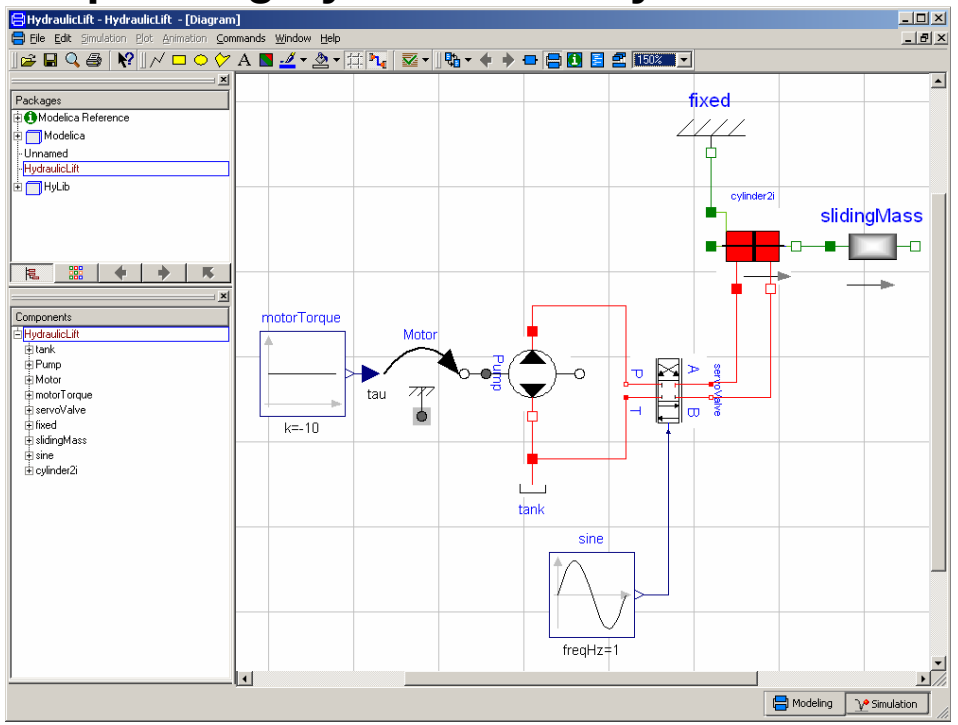
Composition of Behavior Model

System Architecture

SysML Internal Block Diagram



Corresponding System-level Dynamic Model



Corresponding Algebraic Model

slidingMass.vel = vmax;

slidingMass.vel = cylinder.piston.vel;

cylinder.portA.q = cylinder.piston.vel*cylinder.area;

cylinder.portA.q = pump.portP.q;

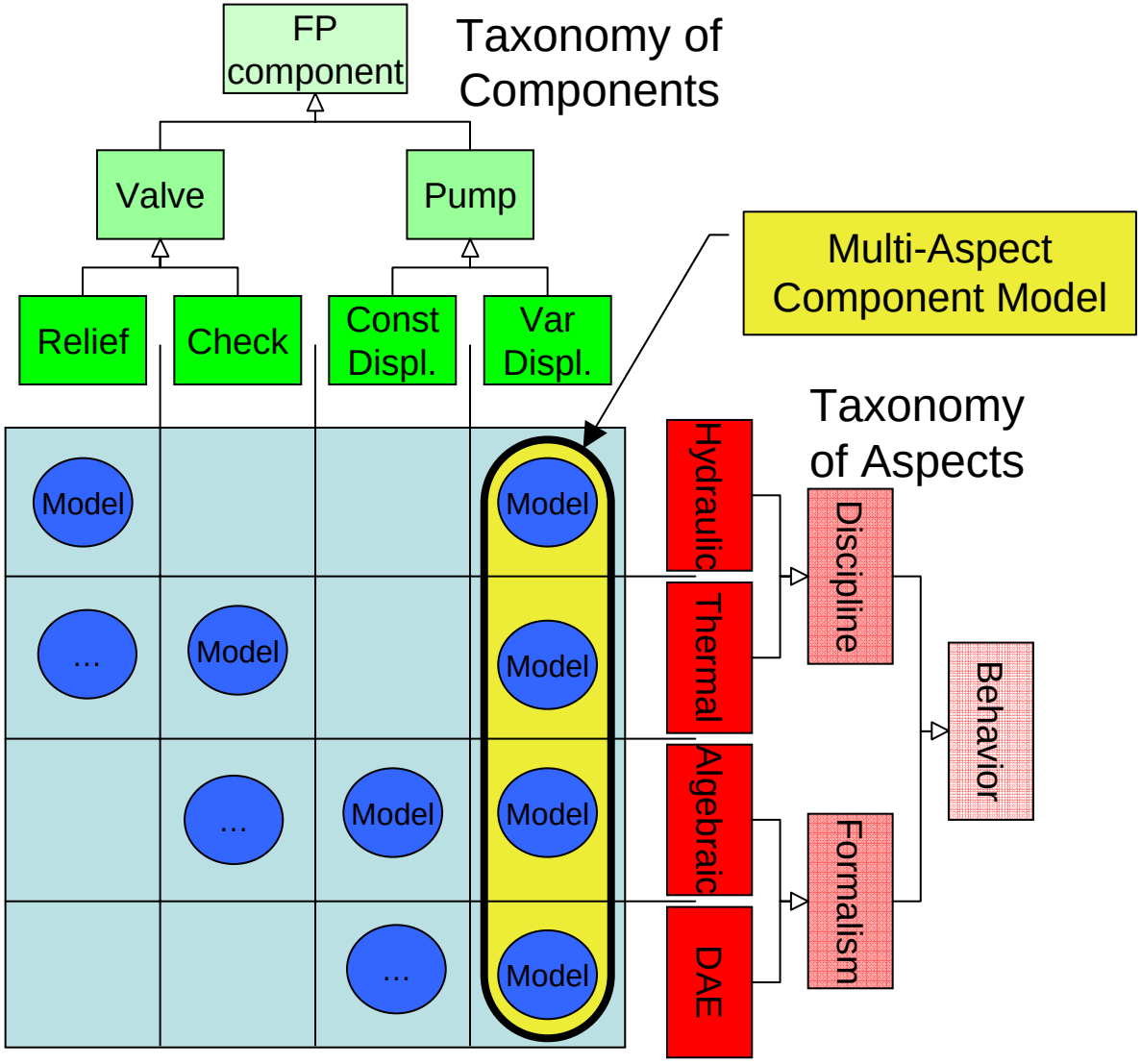
pump.shaft.omega*pump.displ / (2*pi) = pump.portP.q;

motor.shaft.omega = pump.shaft.omega;

© 2008, Chris Paredis

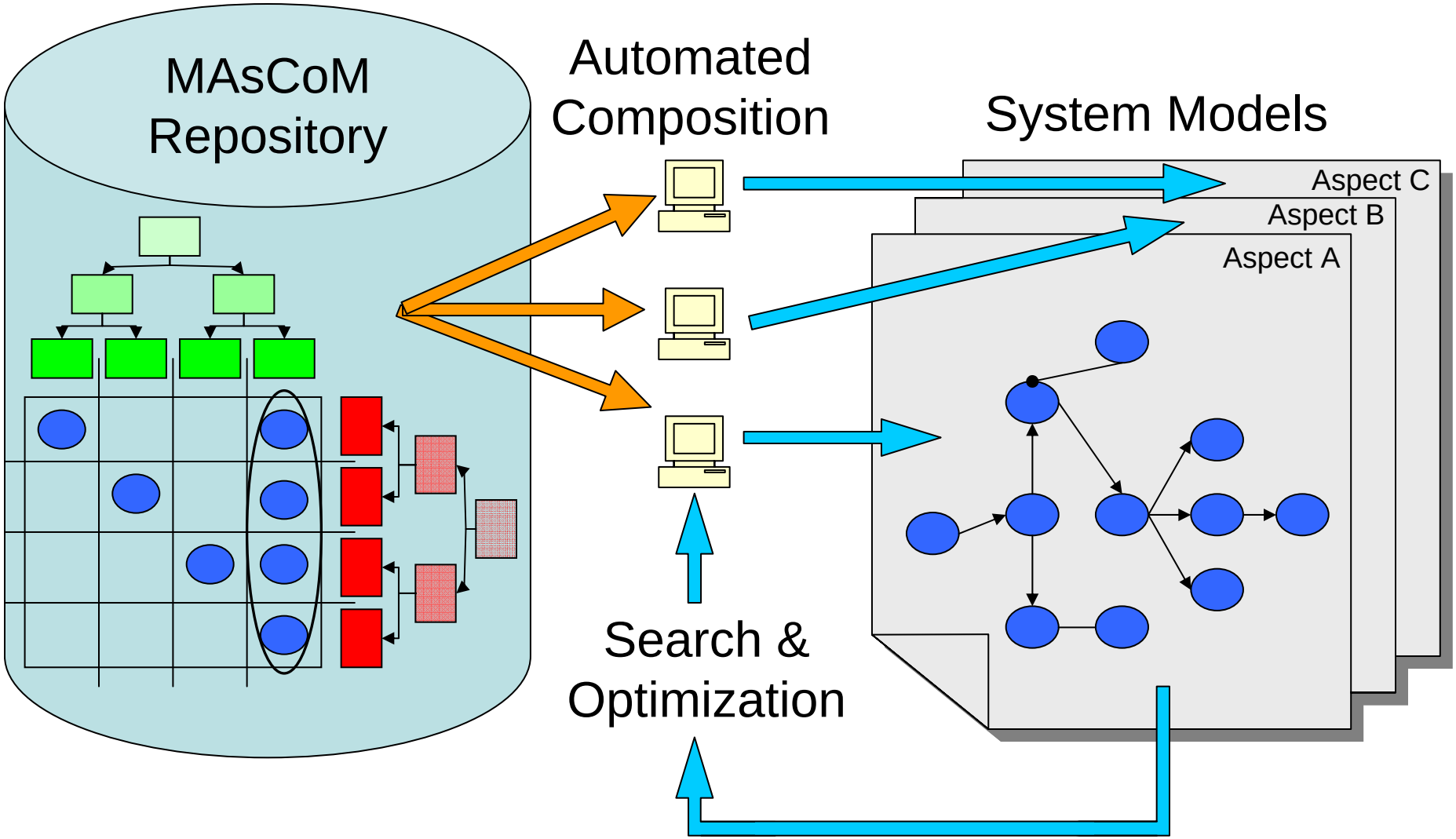
Multi-Aspect Component Models

- MAsCoMs
  - Define relationships between all the models related to a particular component or subsystem
  - Model context
- Aspects
  - Discipline
  - Lifecycle phase
  - Discretization
    - Time
    - Space
  - Math Formalism
  - Representation Syntax

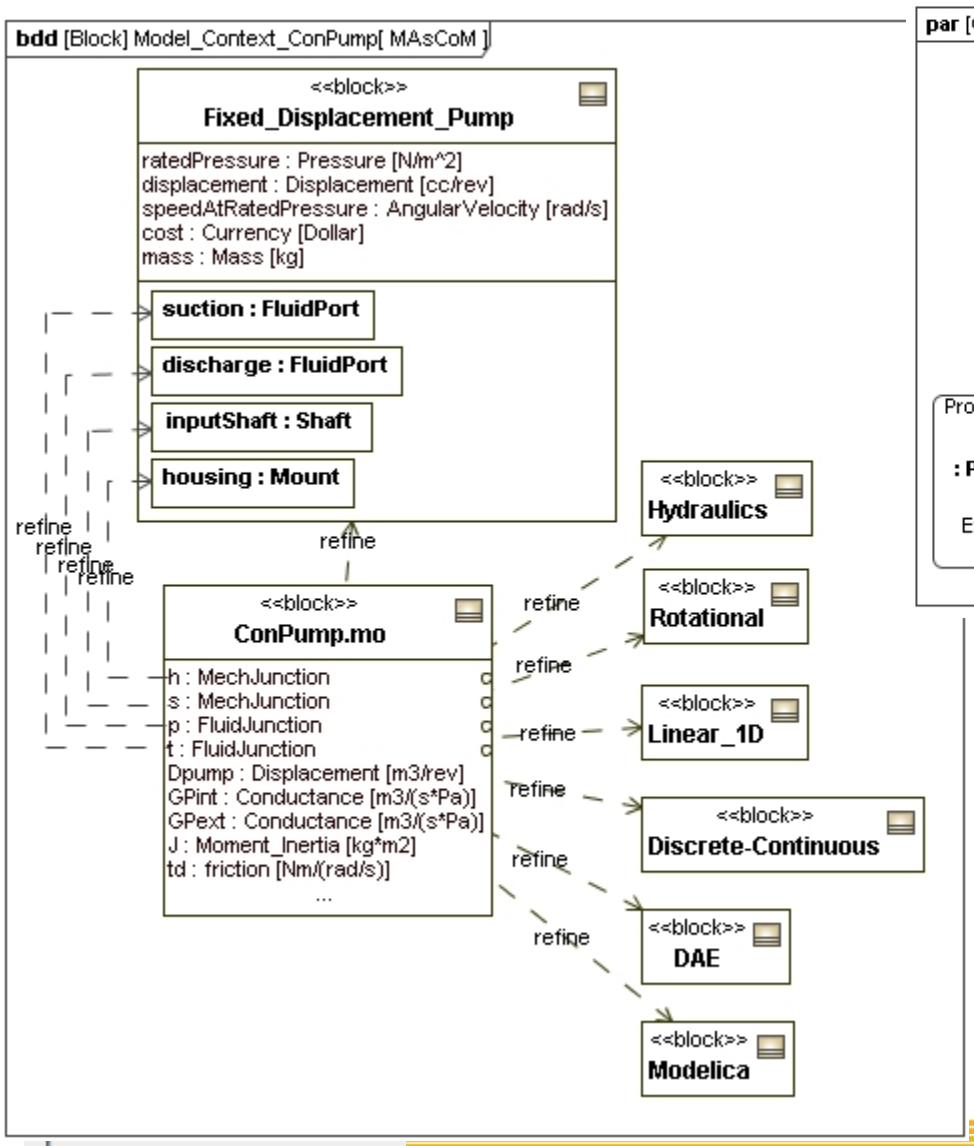


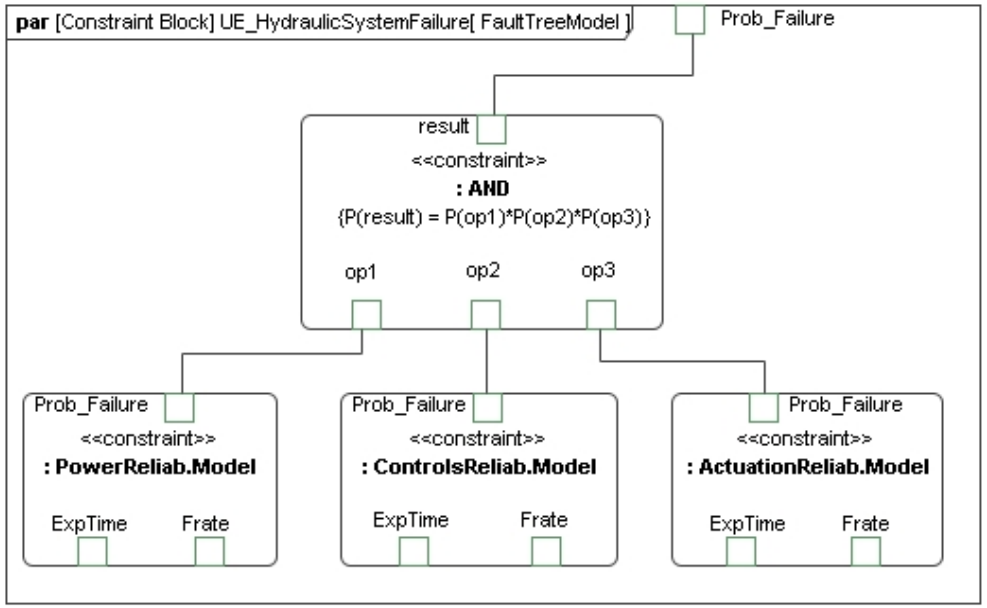
© 2008, Chris Paredis

Composition Process



Multi-Aspect Component Models in SysML





Composition of reliability models

Characterization of analysis model of a component

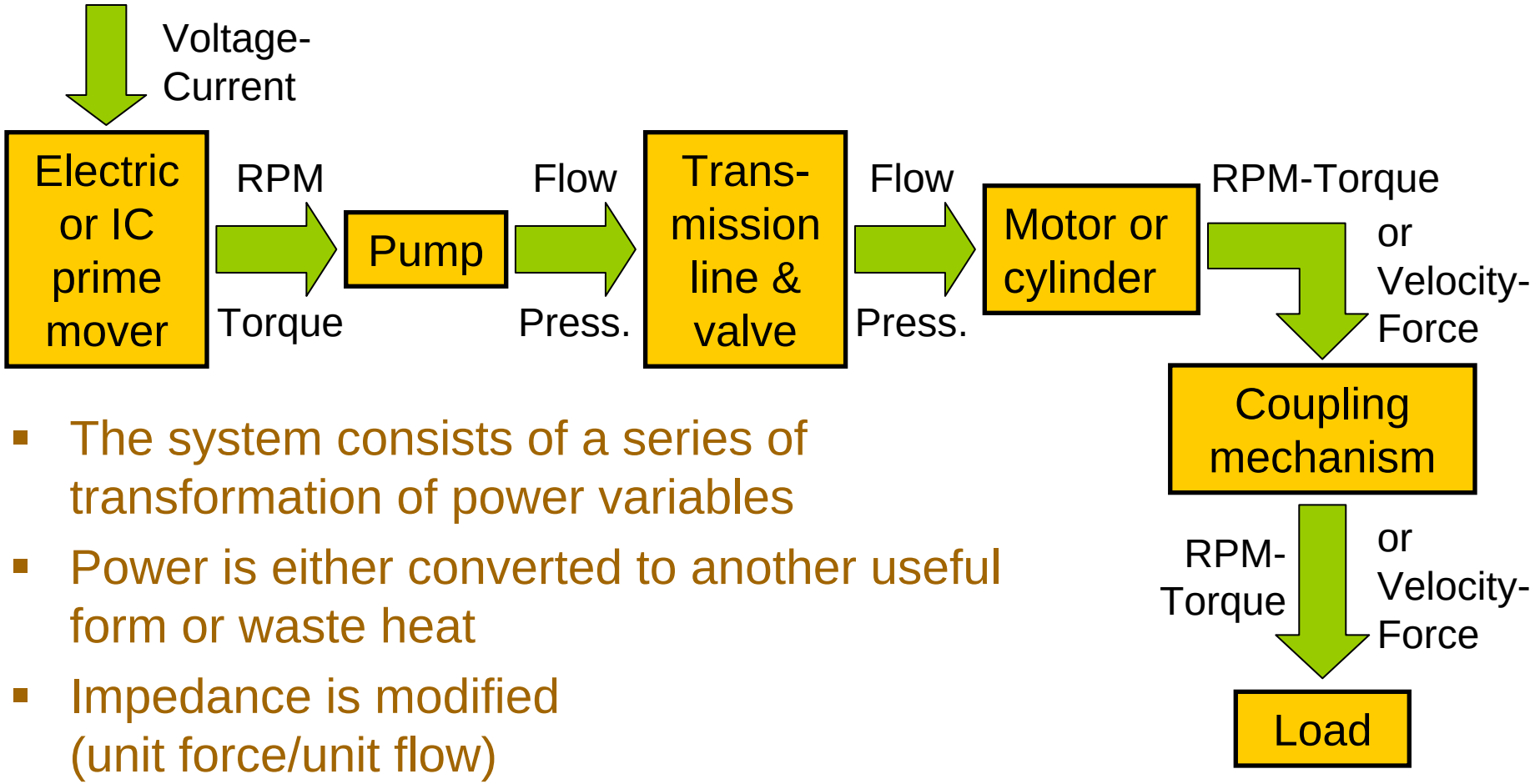
© 2008, Chris Paredis

Presentation Overview

- What is Model-Based Systems Engineering?
  Why MBSE?
- A simple model of knowledge and information in MBSE
- Which kinds of knowledge and information?
  - Information schema
  - Language mappings
  - Abstractions and idealizations
  - Analysis patterns
  - Synthesis patterns
  - Design Workflow
- Challenges and Summary

© 2008, Chris Paredis

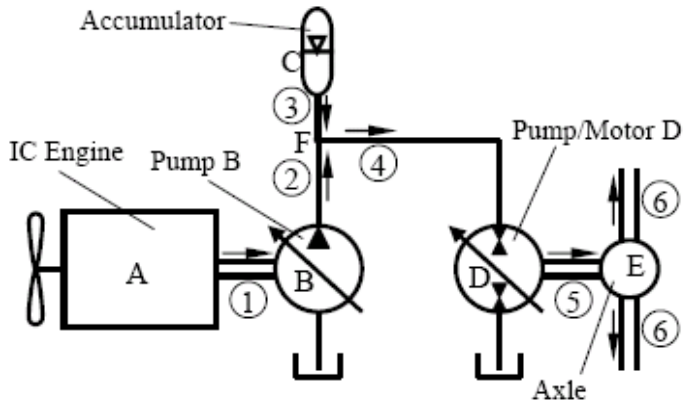
System Synthesis Patterns

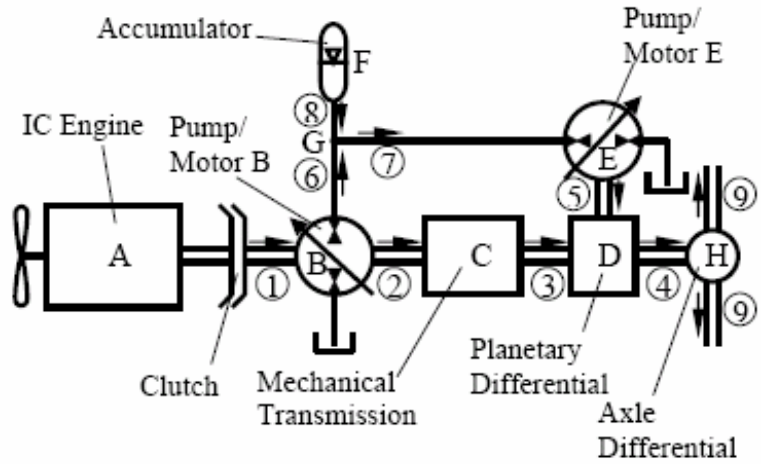


- The system consists of a series of transformation of power variables
- Power is either converted to another useful form or waste heat
- Impedance is modified (unit force/unit flow)
- Power is controlled
- Function is achieved

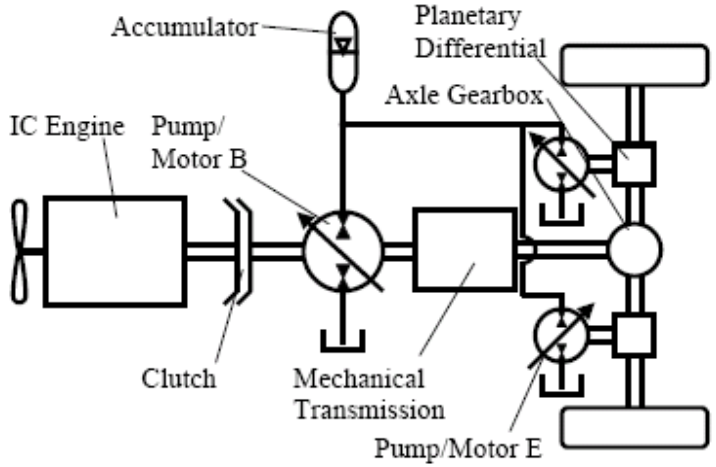
© 2008, Chris Paredis

Different Architectures





- Key issues:
  - Hardware + Controllers
  - Compatibility constraints
  - Hidden dependencies
  - Best solution depends on the preferences of the designer
  - Each concept is a large set of design alternatives



© 2008, Chris Paredis

Graph Grammars

- Capture application domain knowledge in a formal grammar
- Grammar describes *plausible* configurations
  - Ideally optimal configurations
  - Even feasible configurations may too complex to define in grammar
- Explore design space by applying transformation rules in some randomized fashion
- Which knowledge to model in terms of transformations? Which knowledge to model in terms of analyses?

© 2008, Chris Paredis

Capturing Fluid Power Knowledge in Graph Transformations

Requirements & Objectives — SysML

Topology Generation using Graph Transf

System Alternatives

MAsCoMs — SysML

Model Composition using Graph Transf

System Behavior Models — SysML

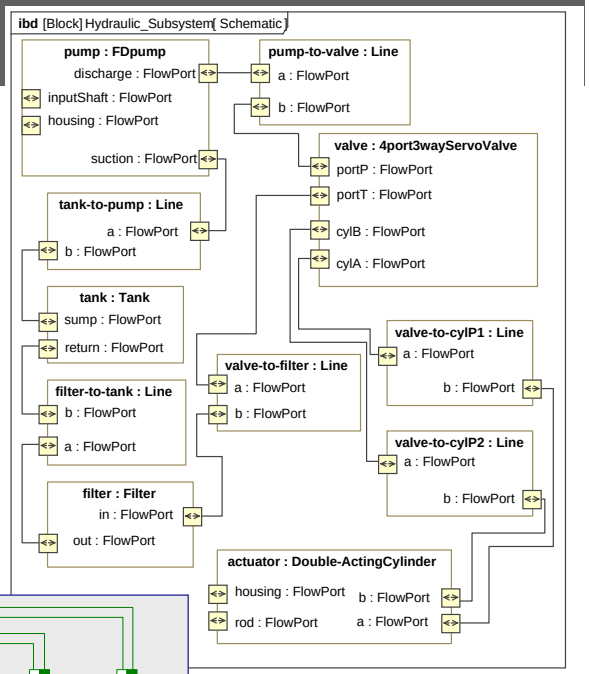
Model Translation using Graph Transf

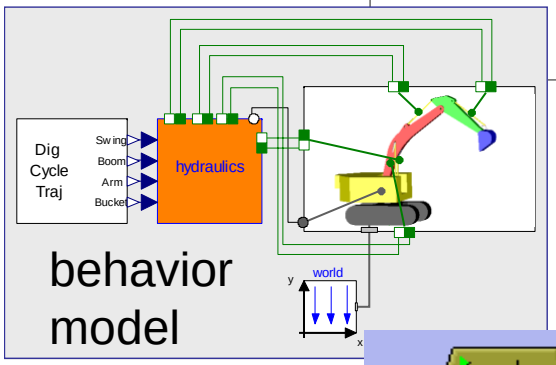
Executable Simulations — Dymola

Simulation Configuration using Graph Transf

Design Optimization — ModelCenter

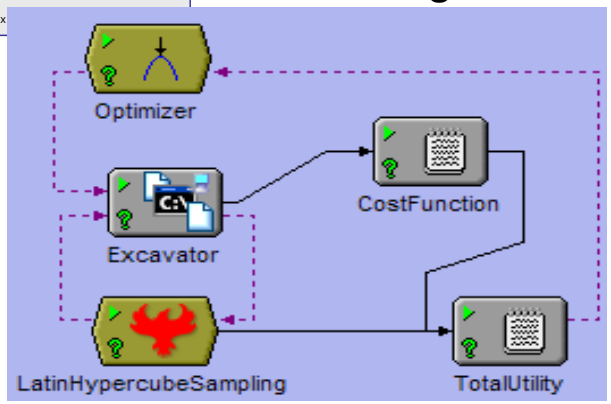
system alternative





behavior model

simulation configuration



Design Workflow

- Ideally all models would always be consistent
- In practice, changes need to be propagated carefully
  - e.g., large parametric assembly model, ECAD-MCAD integration
  - Traceability is crucial, but automated propagation is not desirable
- Propagation is controlled at decision points
  - Human decision-maker needs to remain in the loop
  - But in the end, all models should be consistent
- How does one model the application of transformations?

© 2008, Chris Paredis

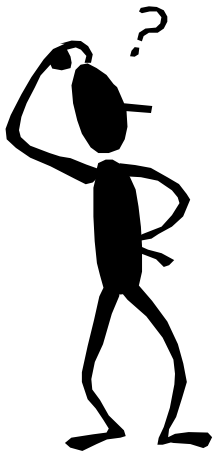
Challenges and Summary

- Evolving representations
  → maintenance of transformations, schema, instances
- Maintaining multiple views – bi-directional
- Modeling idealizations as graph transformations
- Capturing knowledge context of synthesis rules
- Hidden dependencies
  → system-level interfaces are only models
- Workflow and consistency management

© 2008, Chris Paredis

Acknowledgments

- Graduate Students
  - Manas Bajaj
  - Jonathan Jobe
  - Tommy Johnson
  - Aleks Kerzhner
  - Richard Malak
  - Roxanne Moore
  - Stephanie Thompson
  - Lina Tucker
- Collaborators
  - Roger Burkhart (Deere)
  - Sandy Friedenthal (LMCO)
  - Leon McGinnis (GT)
  - Russell Peak (GT)
- Sponsors:
  - National Science Foundation Grant DMI-0522116
  - National Science Foundation, Center for Complex and Efficient Fluid Power
  - Deere & Co.
  - Lockheed Martin



Questions?

© 2008, Chris Paredis