The OMG SysML-Modelica Transformation Specification:

Supporting Model-Based Systems Engineering with SysML and Modelica

Chris Paredis
Associate Director
Model-Based Systems Engineering Center
Georgia Tech
Chris.paredis@me.gatech.edu

www.pslm.gatech.edu/courses
www.omg.org/ocsmp

SysML-Modelica Transformation Specification: Context & Objective

- Two complementary languages for Systems Engineering:
  - Descriptive modeling in SysML
  - Formal equation-based modeling for analyses and trade studies in Modelica

- Objective:
  - Leverage the strengths of both SysML and Modelica by integrating them to create a more expressive and formal MBSE language.
  - Define a formal Transformation Specification:
    - » a SysML4Modelica profile
    - » a Modelica abstract syntax metamodel
    - » a mapping between Modelica and the profile

Presentation Overview

- What is SysML?
- What is Modelica?

- Motivating Example: Design & Analysis of Robot

- SysML-Modelica Transformation Specification
- Transformations in Model-Based Systems Modeling
- Summary

Acknowledgements

Working Group Members

- Yves Bernard (EADS)
- Roger Burkhart (Deere & Co)
- Wuzhu Chen (Univ. Braunschweig)
- Hans-Peter De Koning (ESA)
- Sandy Friedenthal (Lockheed Martin)
- Peter Fritzson (Linköping University)
- Nerijus Jankevicius (No Magic)
- Alek Kerzhner (Georgia Tech)
- Andreas Korff (Atego)
- Chris Paredis (Georgia Tech)
- Axel Reichwein (Georgia Tech)
- Nicolas Rouquette (JPL)
- Wladimir Schamai (EADS)

Students / post-docs

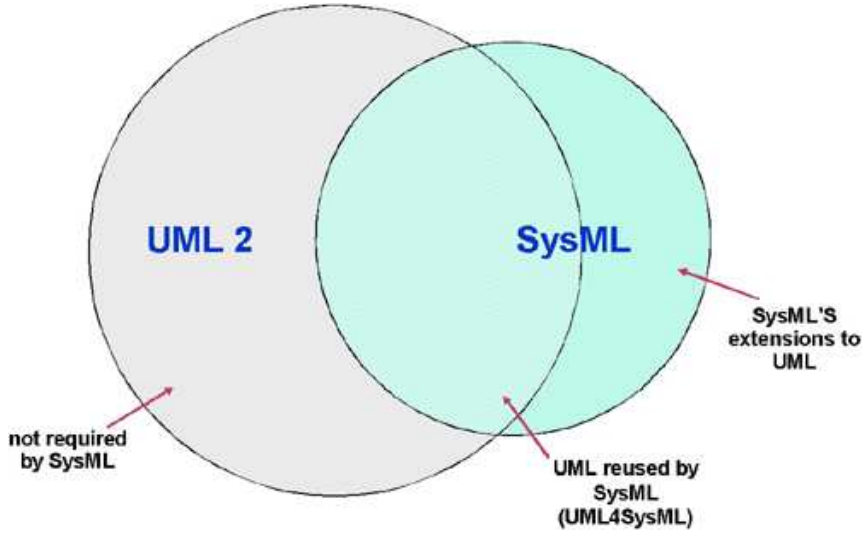
- Kevin Davies
- Sebastian Herzig
- Alek Kerzhner
- Ben Lee
- Roxanne Moore
- Axel Reichwein
- Wladimir Schamai

Sponsors

- Deere & Co
- Lockheed Martin
- National Science Foundation

Systems Modeling Language: SysML

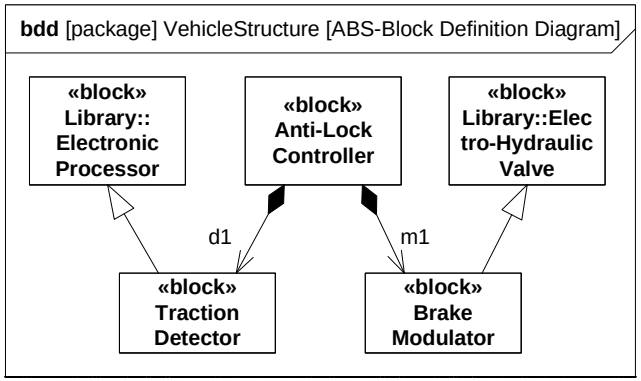
- SysML is an extension of the Unified Modeling Language.
- Supports the modeling of physical systems – not just software
- It is not a methodology or tool
- Improves the ability to exchange systems engineering information among tools and people



(Source: Friedenthal, www.omgsysml.org)

Pillars of SysML: 4 Main Diagram Types

1. Structure



definition

2. Behavior

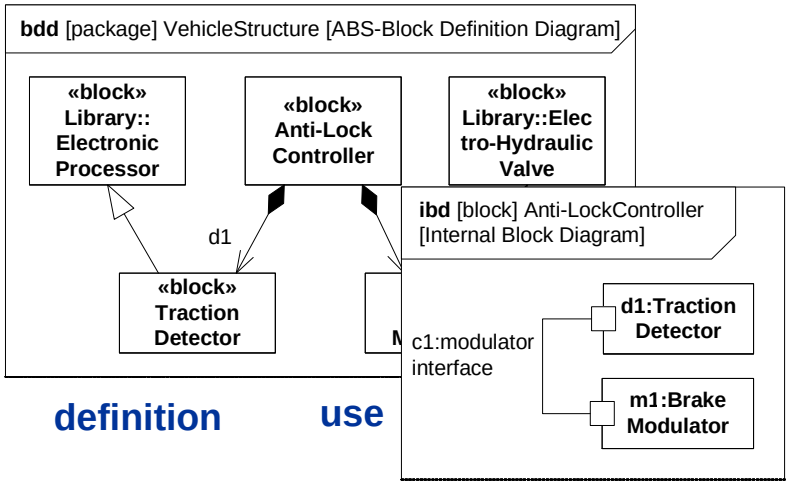
4. Requirements

3. Parametrics

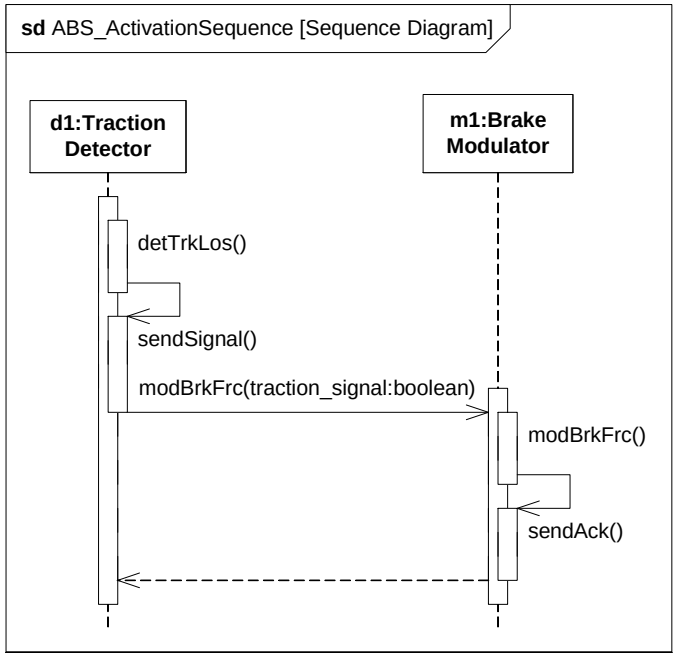
(Source: Friedenthal, www.omgsysml.org)

Pillars of SysML: 4 Main Diagram Types

1. Structure



definition **use**



2. Behavior

interaction

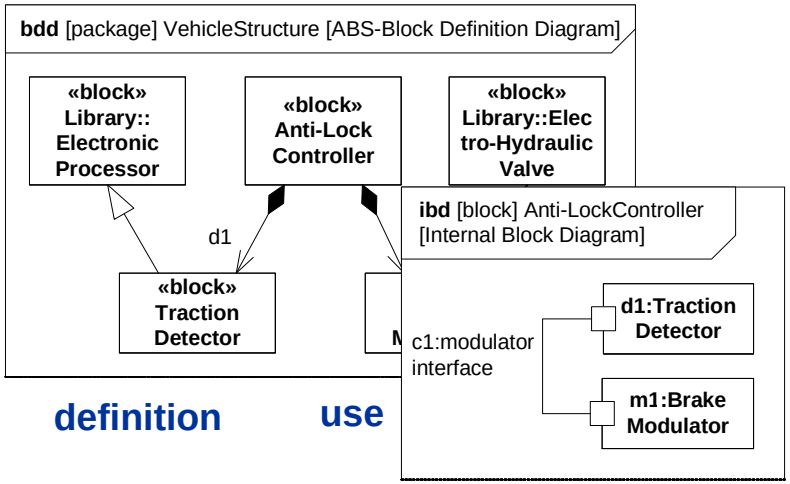
4. Requirements

3. Parametrics

(Source: Friedenthal, www.omgsysml.org)

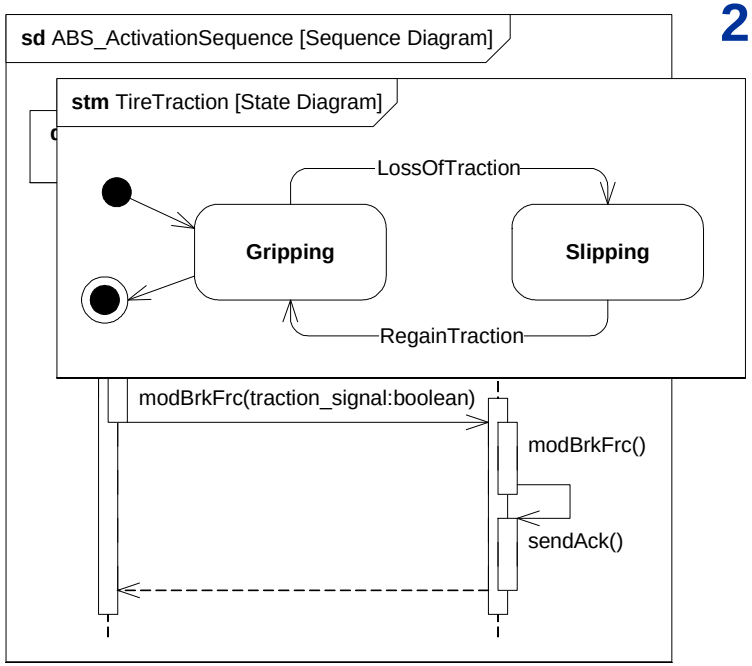
Pillars of SysML: 4 Main Diagram Types

1. Structure



definition **use**

2. Behavior



interaction

state machine

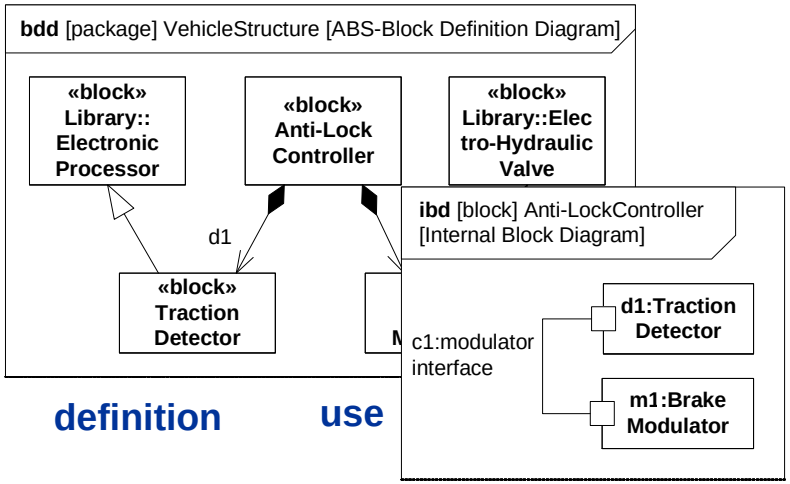
4. Requirements

3. Parametrics

(Source: Friedenthal, www.omgsysml.org)

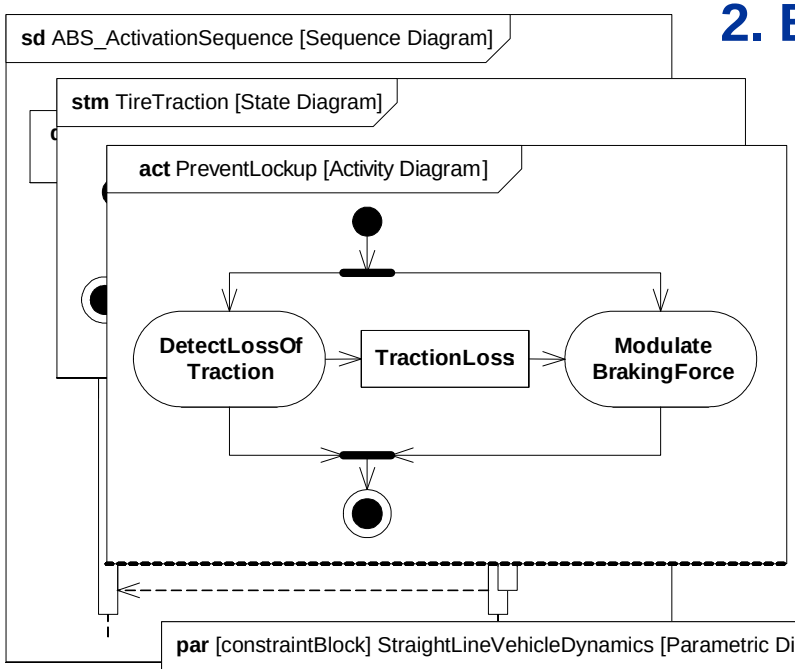
Pillars of SysML: 4 Main Diagram Types

1. Structure



definition **use**

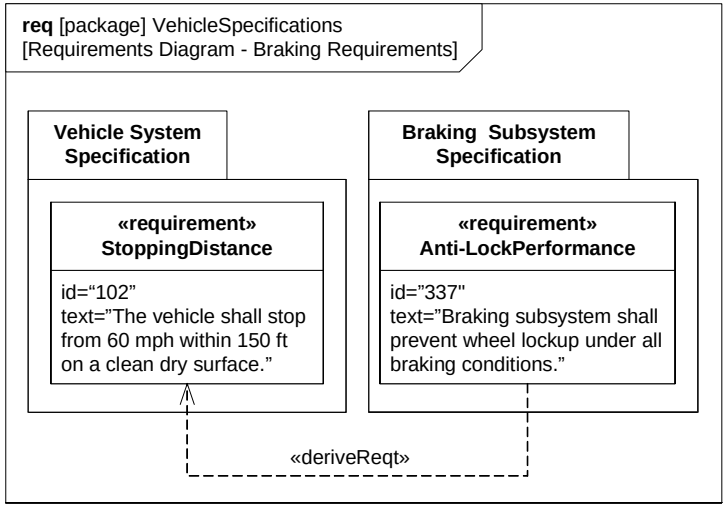
2. Behavior



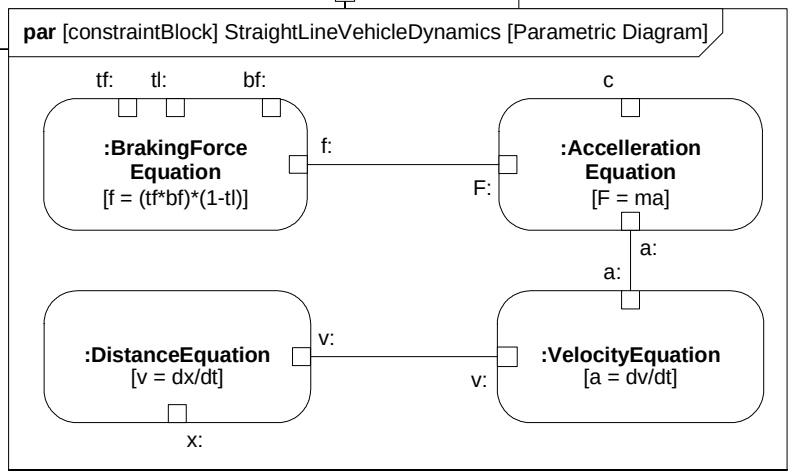
interaction

state machine

activity/ function



4. Requirements

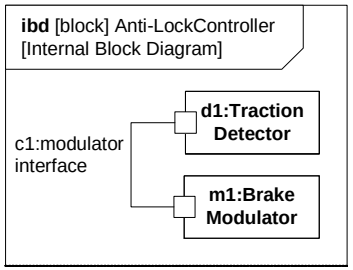


3. Parametrics

(Source: Friedenthal, www.omgsysml.org)

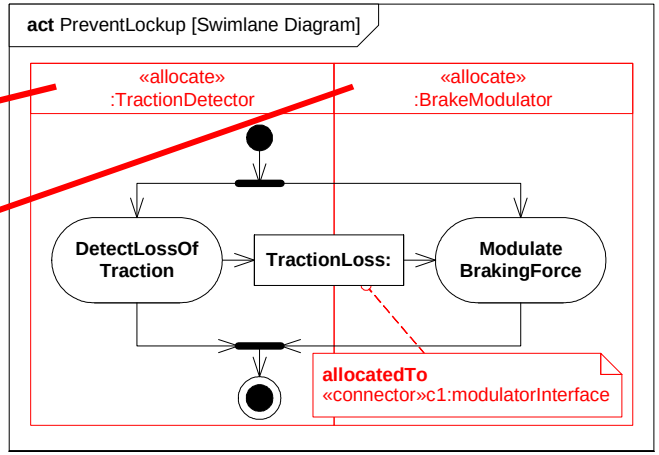
Cross Connecting Model Elements

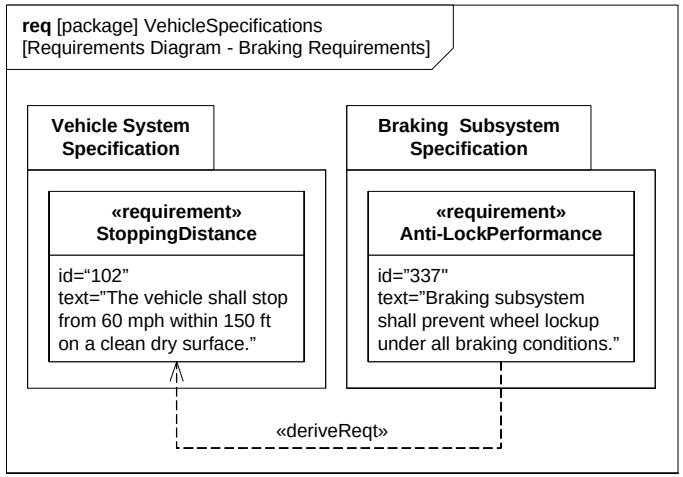
1. Structure

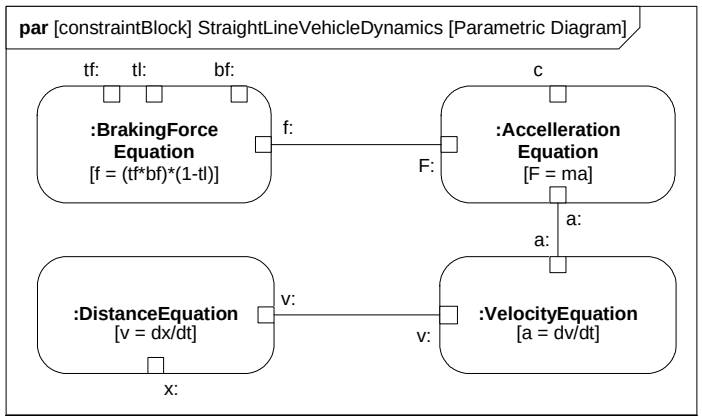


allocate

2. Behavior







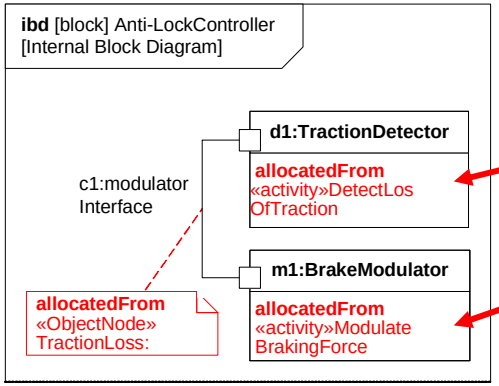
4. Requirements

3. Parametrics

(Source: Friedenthal, www.omgsysml.org)

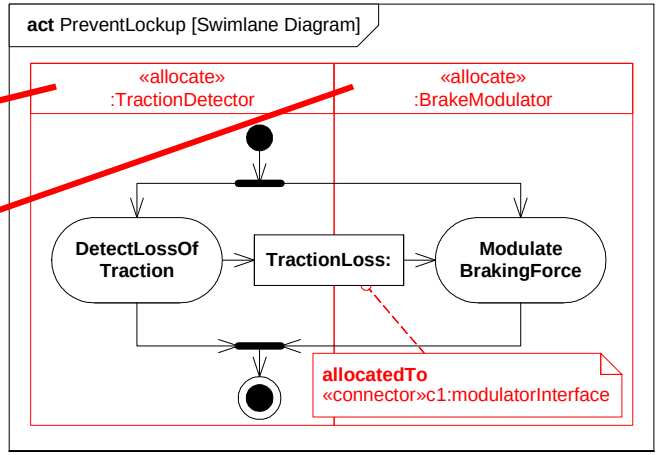
Cross Connecting Model Elements

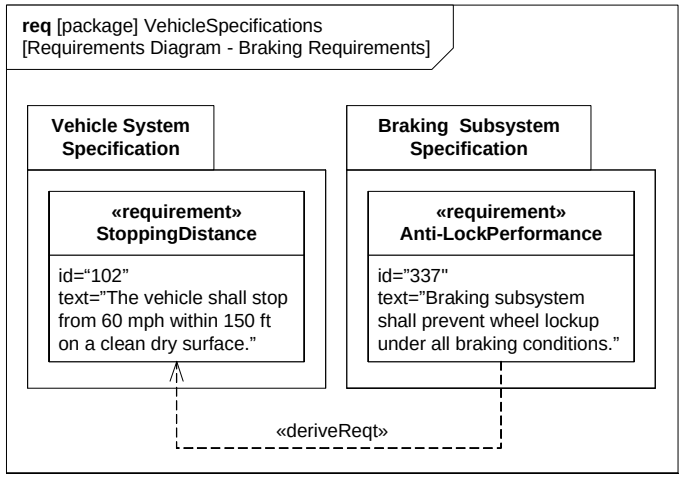
1. Structure

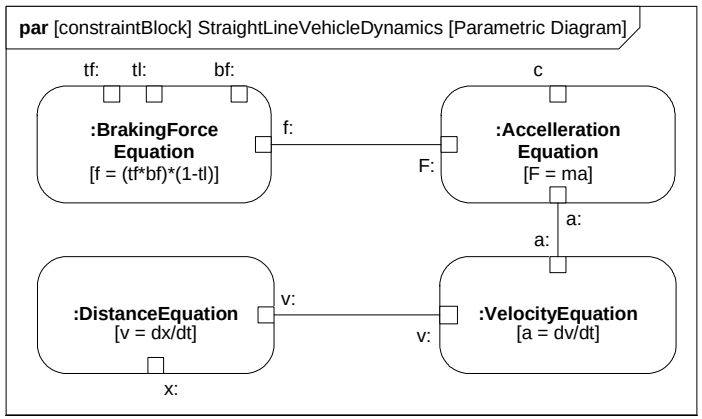


allocate

2. Behavior





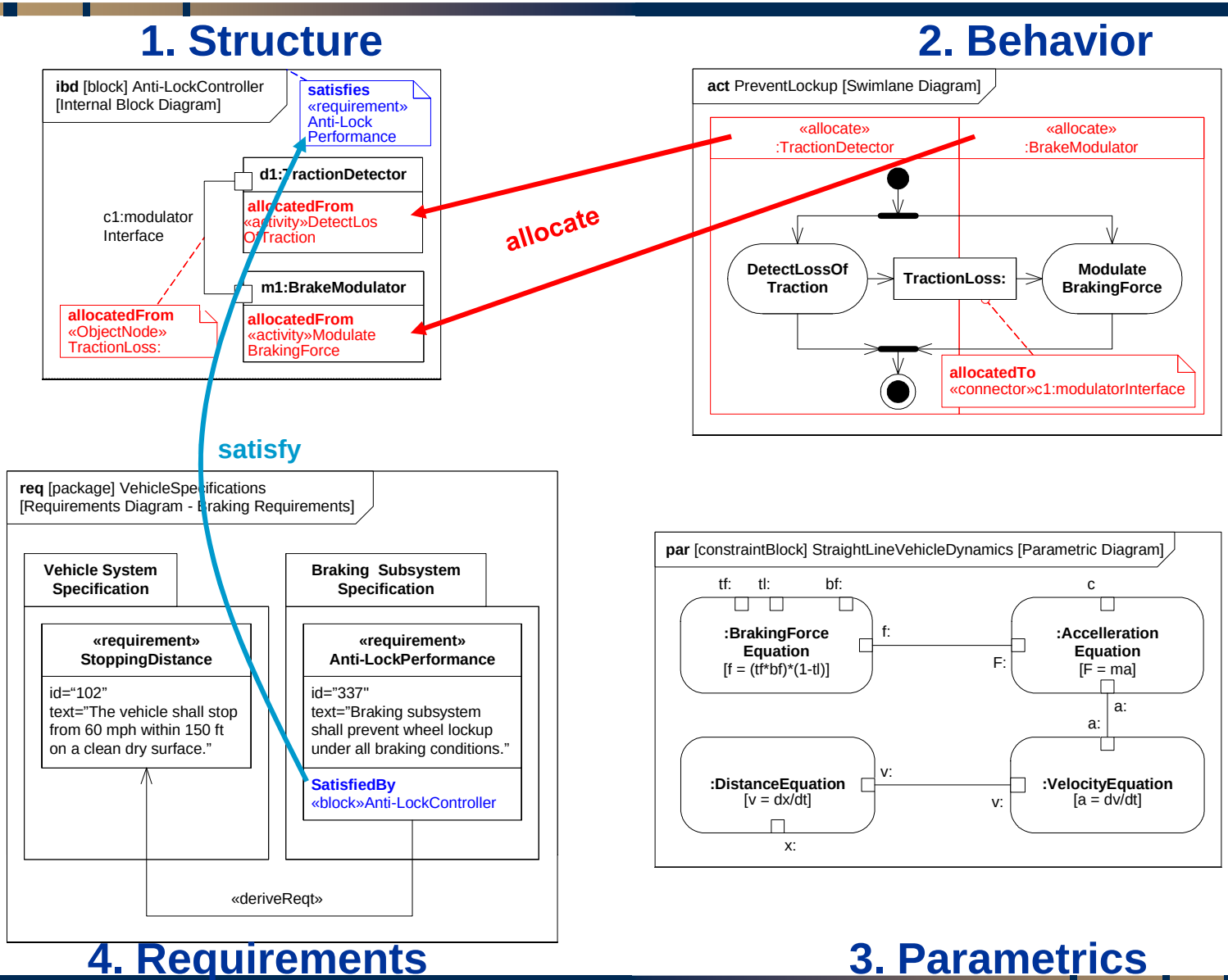


4. Requirements

3. Parametrics

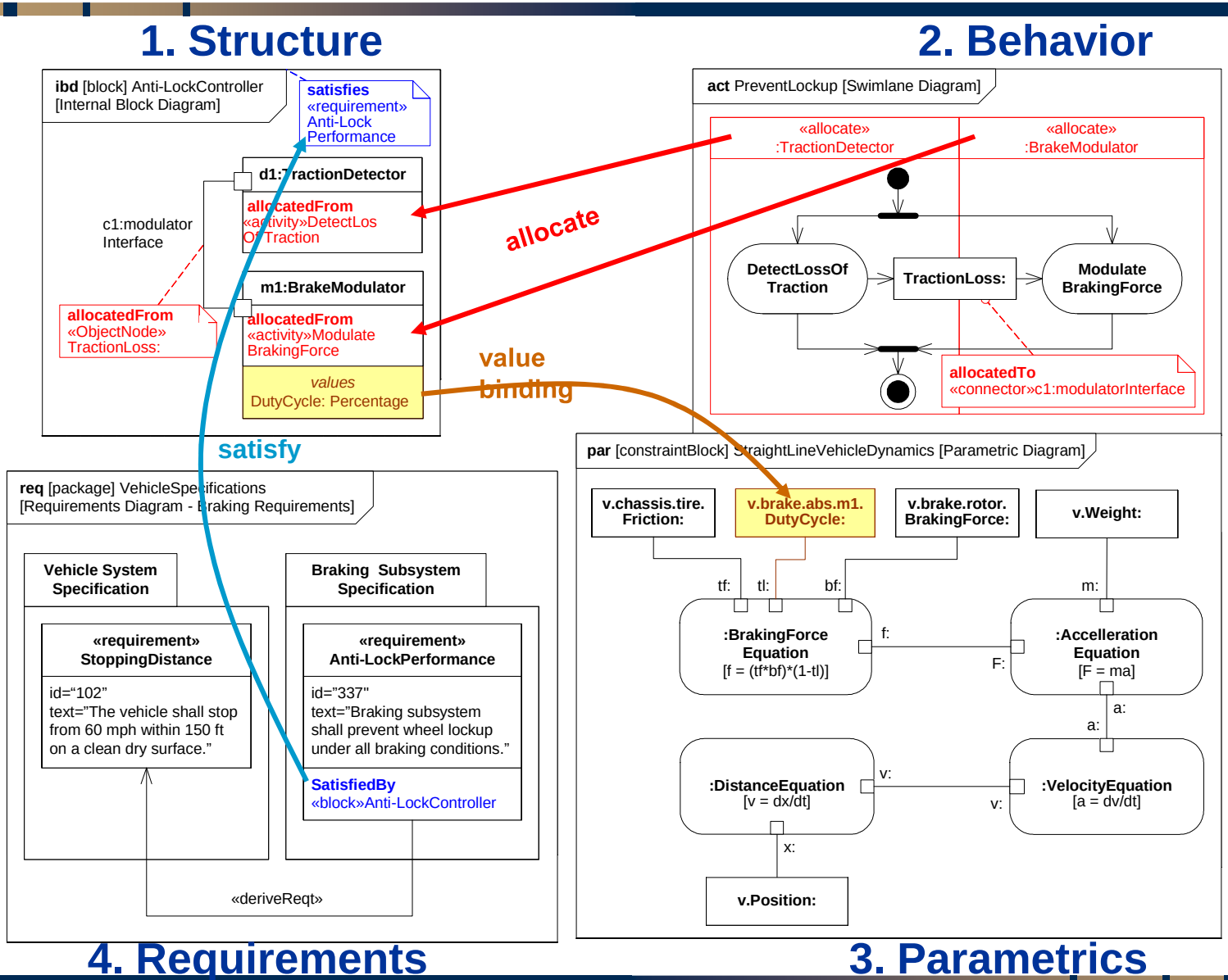
(Source: Friedenthal, www.omgsysml.org)

Cross Connecting Model Elements



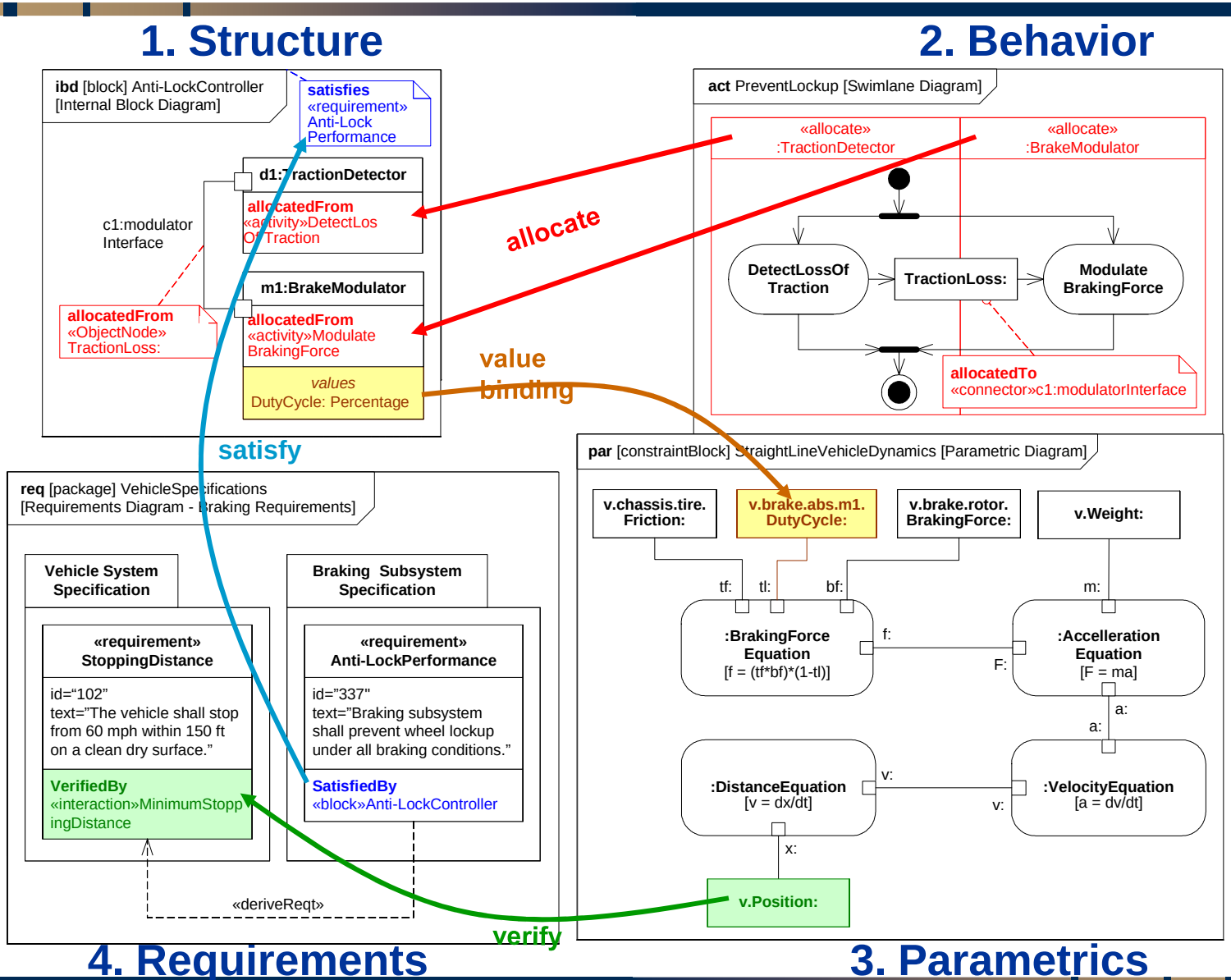
(Source: Friedenthal, www.omgsysml.org)

Cross Connecting Model Elements



(Source: Friedenthal, www.omgsysml.org)

Cross Connecting Model Elements



What is Modelica?
(www.modelica.org)

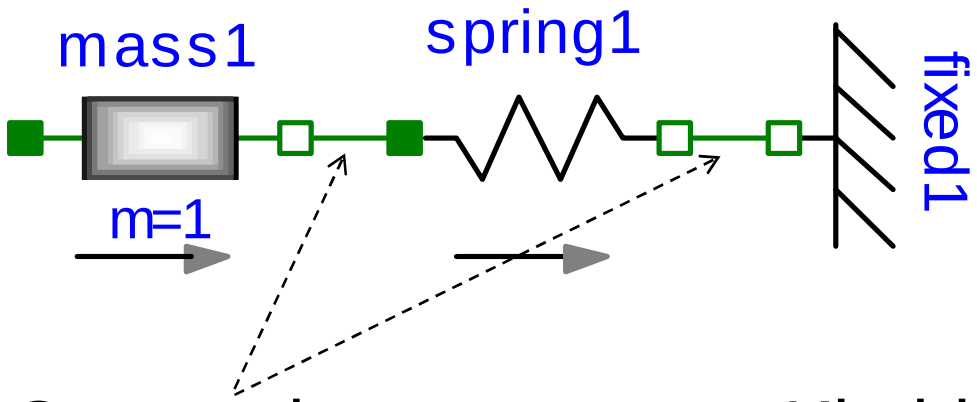
- State-of-the-art Modeling Language for System Dynamics
  - Differential Algebraic Equations (DAE)
  - Discrete Events

- Represents DAE models in an object-oriented, engineering-oriented language
- Multi-(physical)-domain modeling
- Ports represent energy flow (undirected) or signal flow (directed)
- Acausal, equation-based, declarative (f-m*a=0)

Modelica Semantics and Textual Syntax

```
model Spring "Linear 1D translational spring"
  extends Translational.Interfaces.PartialCompliant;
  parameter SI.TranslationalSpringConstant c(final min=0, start = 1)
    "spring constant ";
  parameter SI.Distance s_rel0=0 "unstretched spring length";

equation
  f = c*(s_rel - s_rel0);
end Spring;
```



Graphical symbols defined as annotations in textual models

- Connections represent Kirchhoff semantics
  - Across variables (voltage, pressure,…) are equal
  - Through variables (current, flow rate,…) add to zero

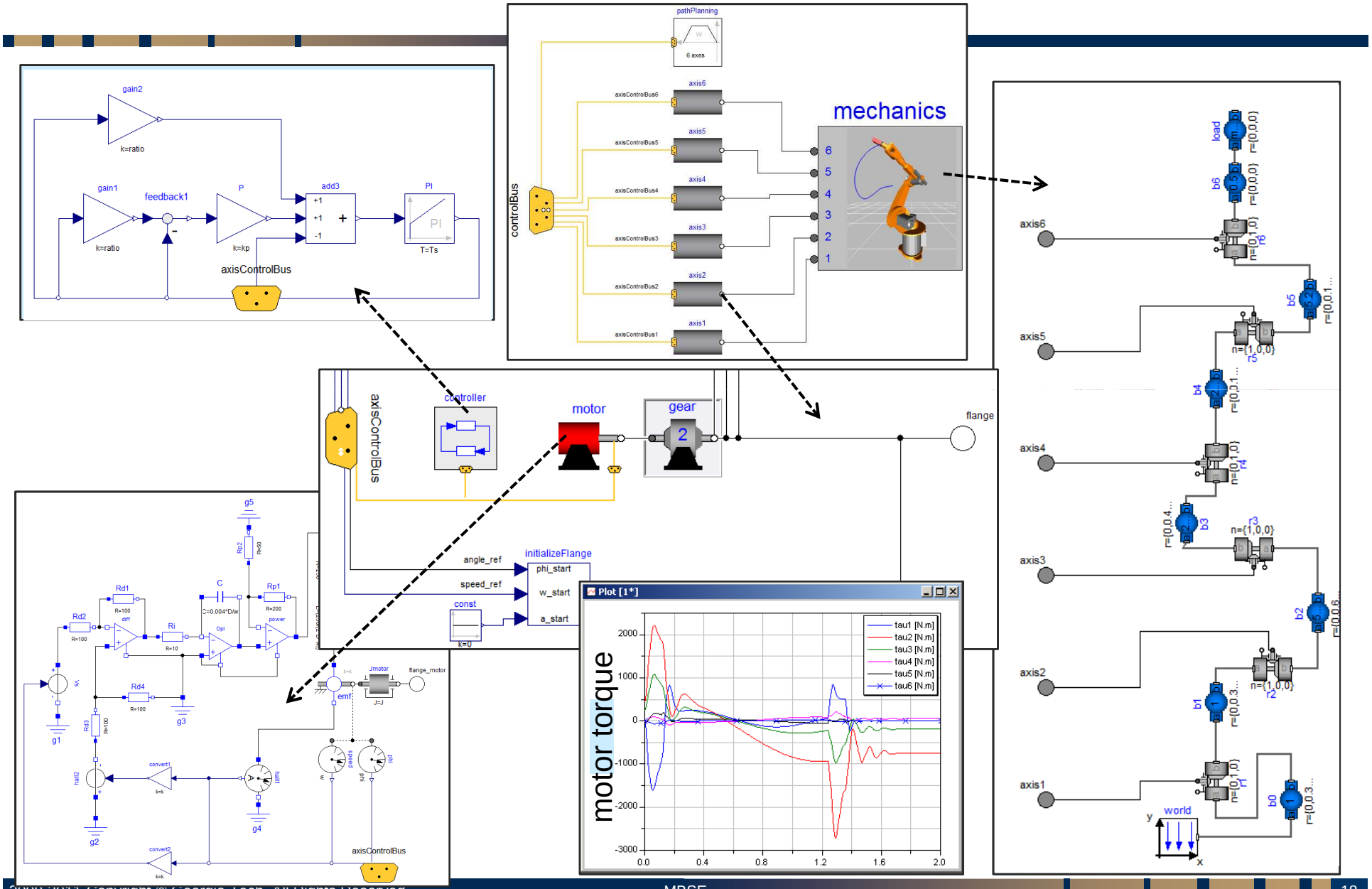
Presentation Overview

- What is SysML?
- What is Modelica?

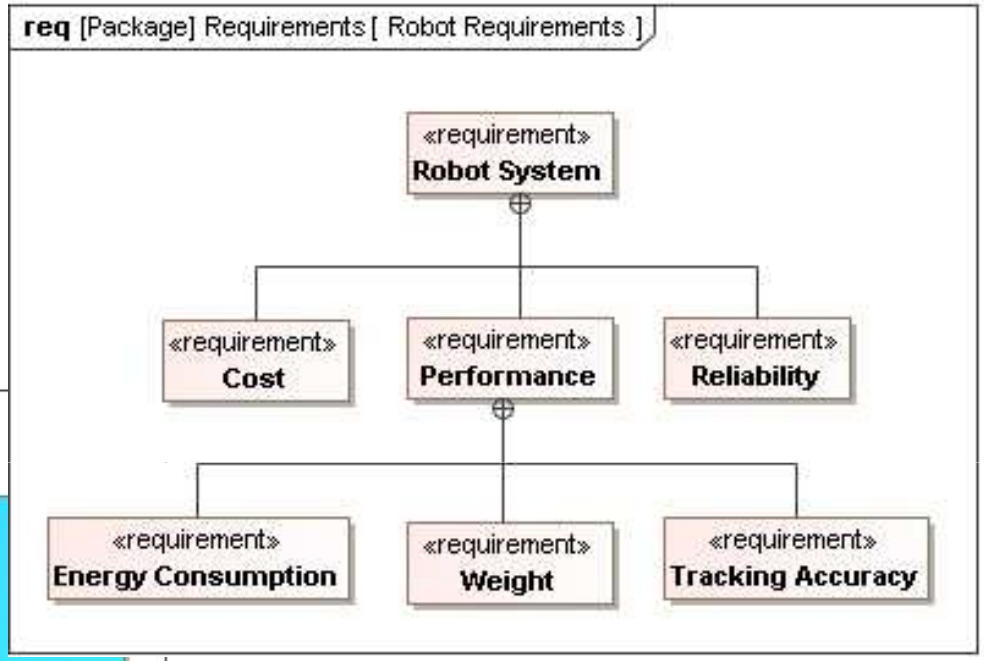
➡ Motivating Example: Design & Analysis of Robot

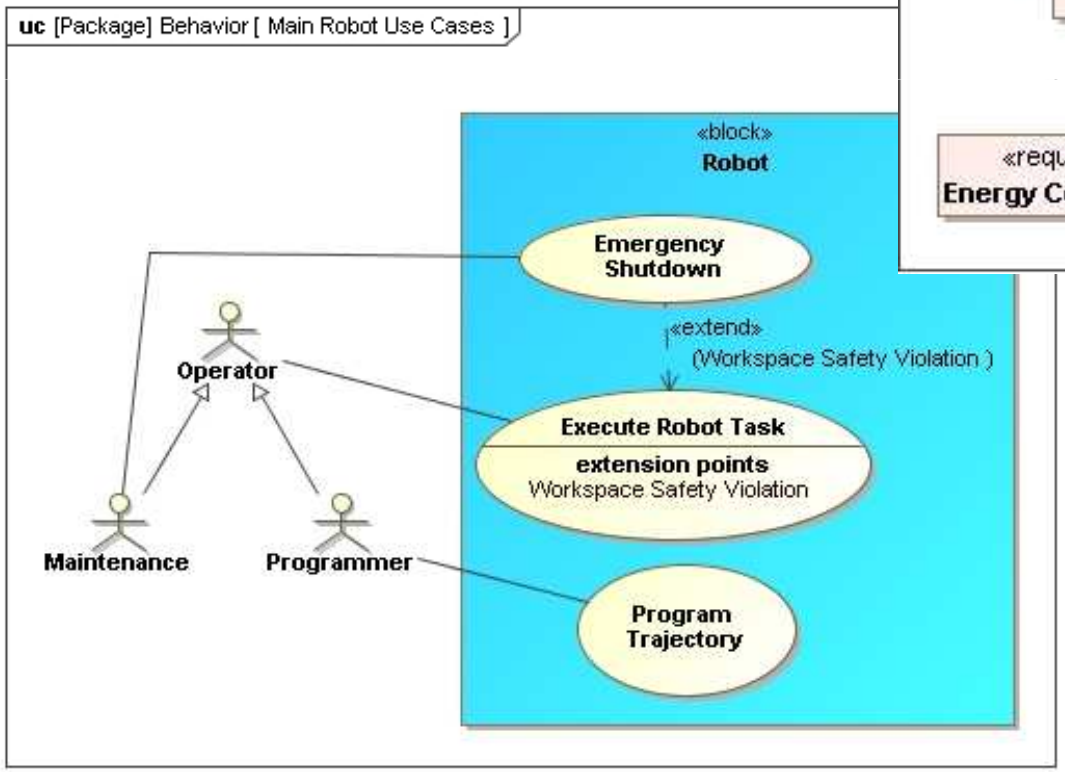
- SysML-Modelica Transformation Specification
- Transformations in Model-Based Systems Modeling
- Summary

A Robot Example in Modelica

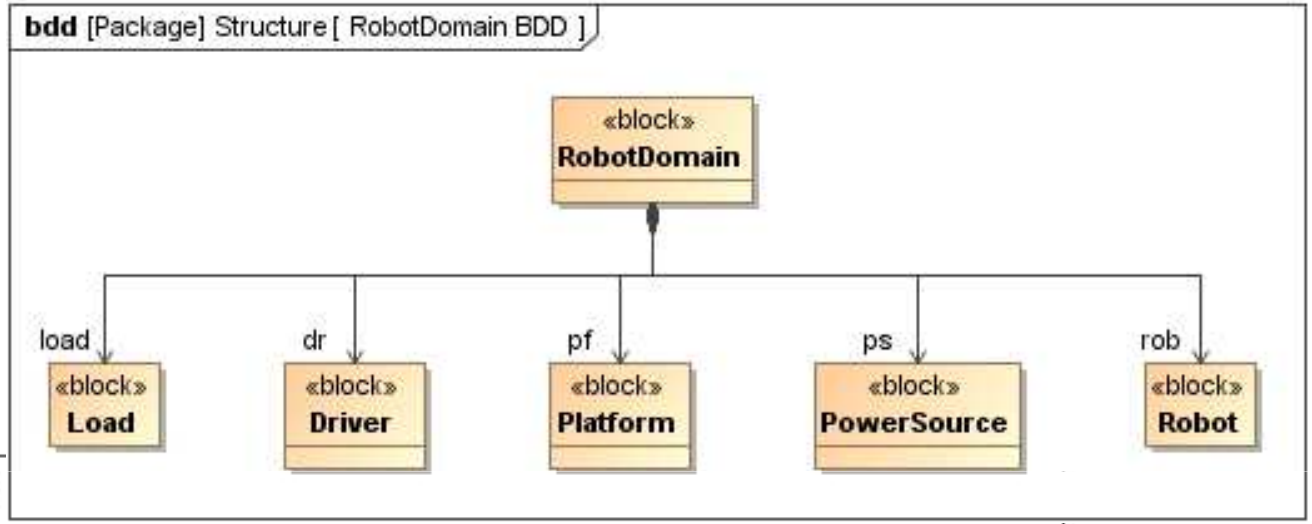


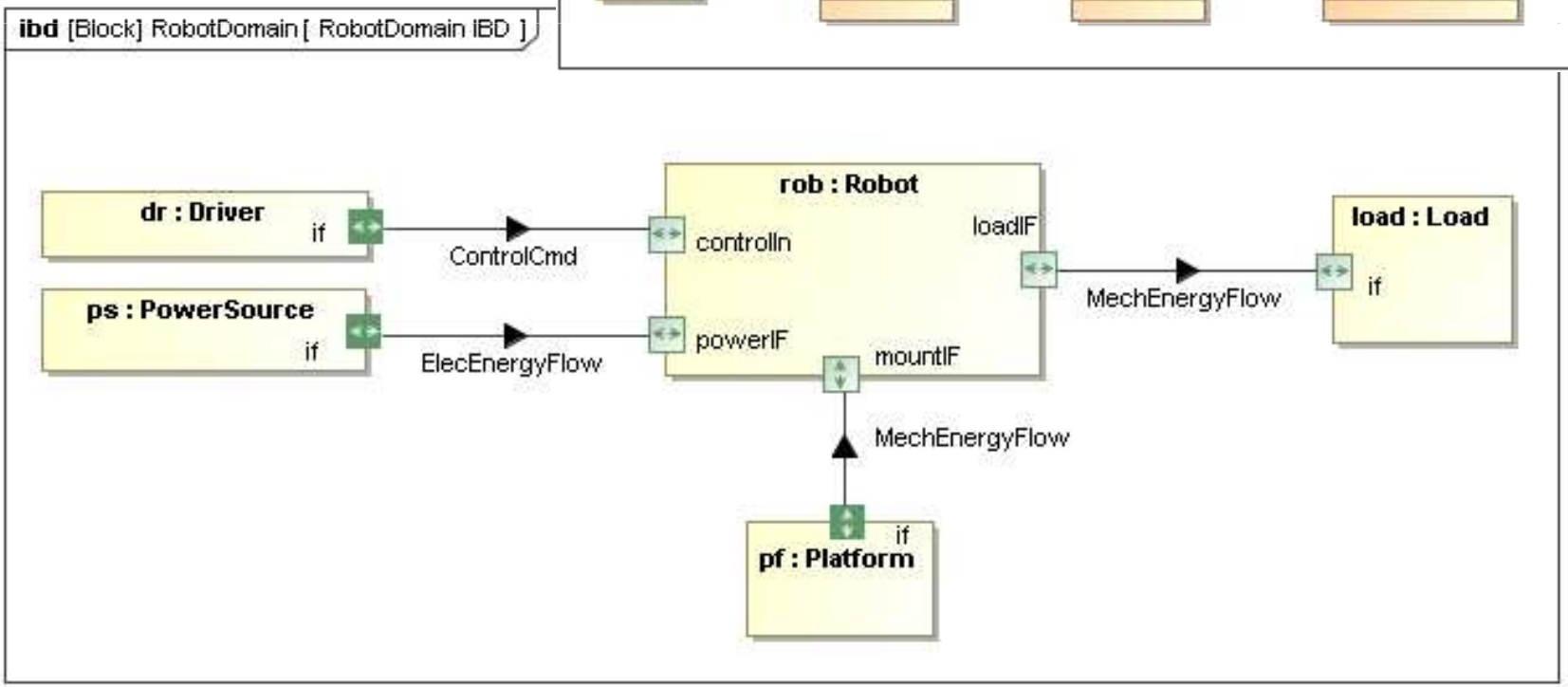
SysML-Modelica Robot Example: UseCases & Requirements



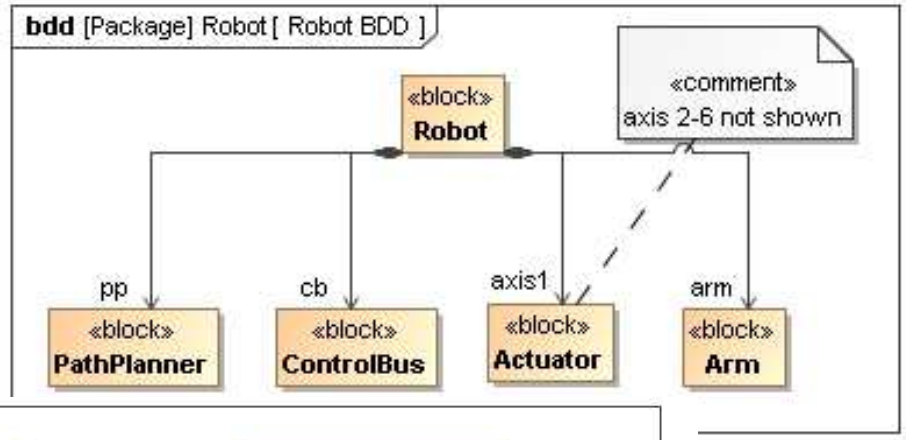


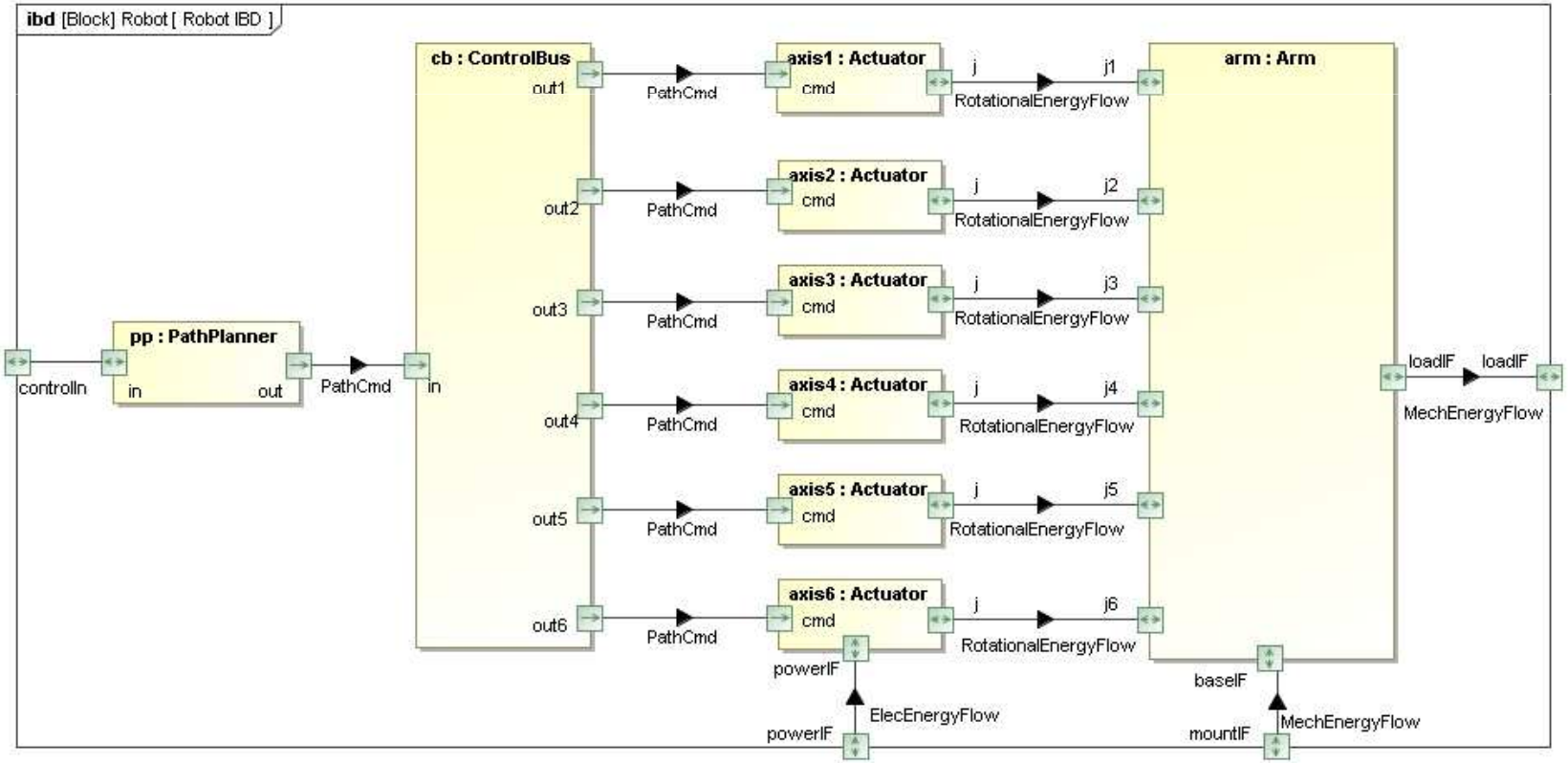
SysML-Modelica Robot Example: Robot Domain BDD & IBD



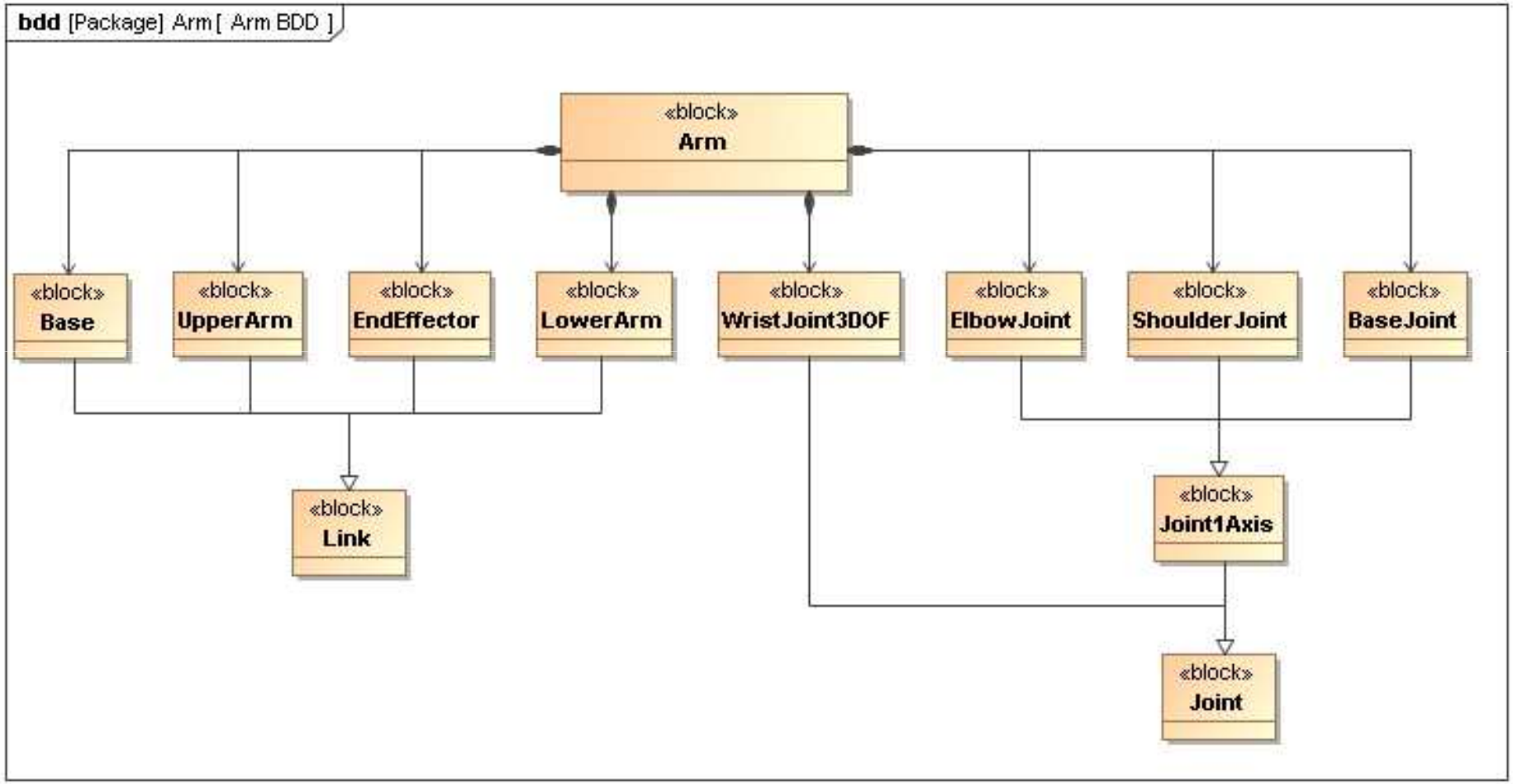


SysML-Modelica Robot Example: Robot BDD & IBD

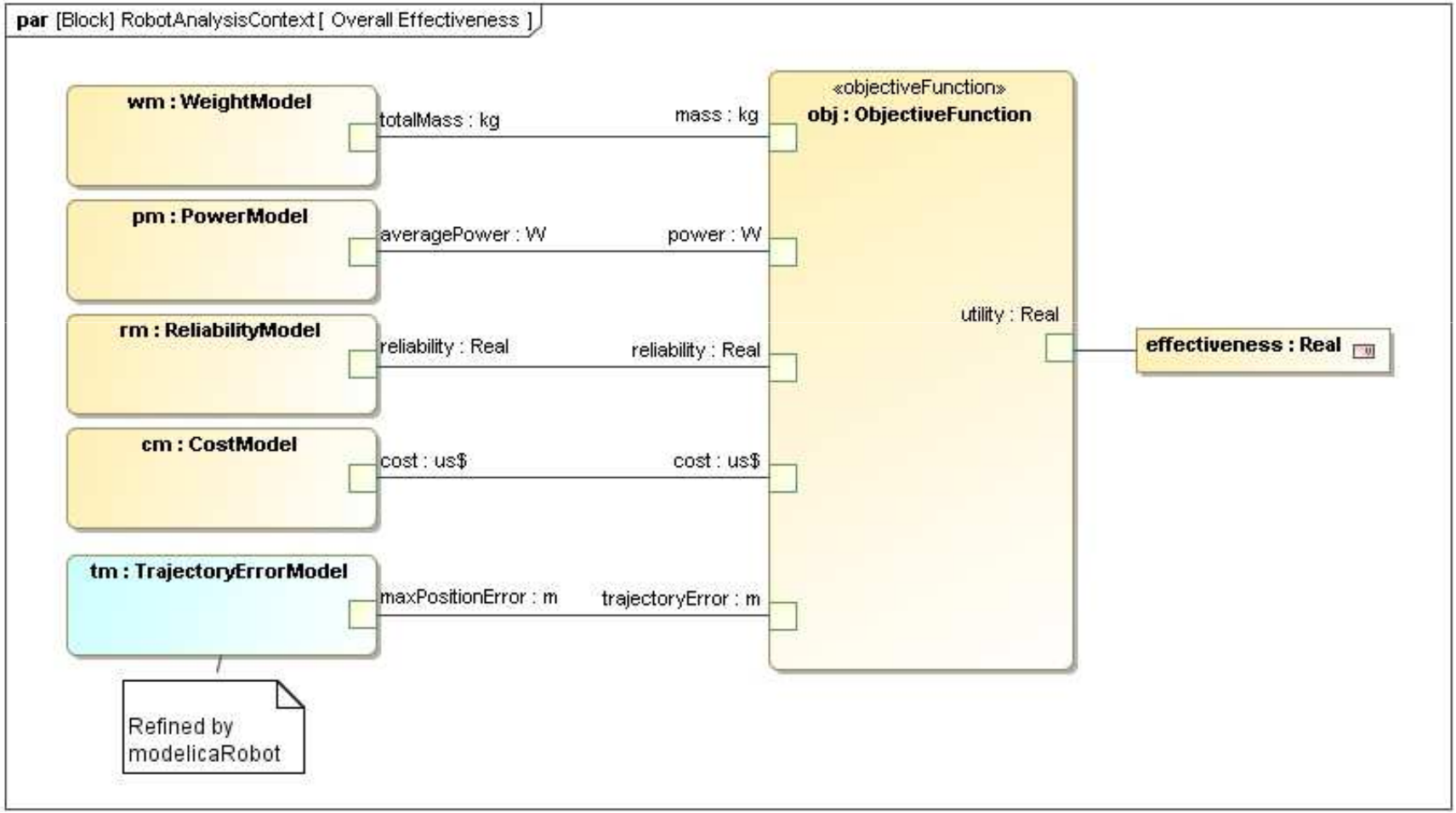




SysML-Modelica Robot Example: Robot Arm BDD

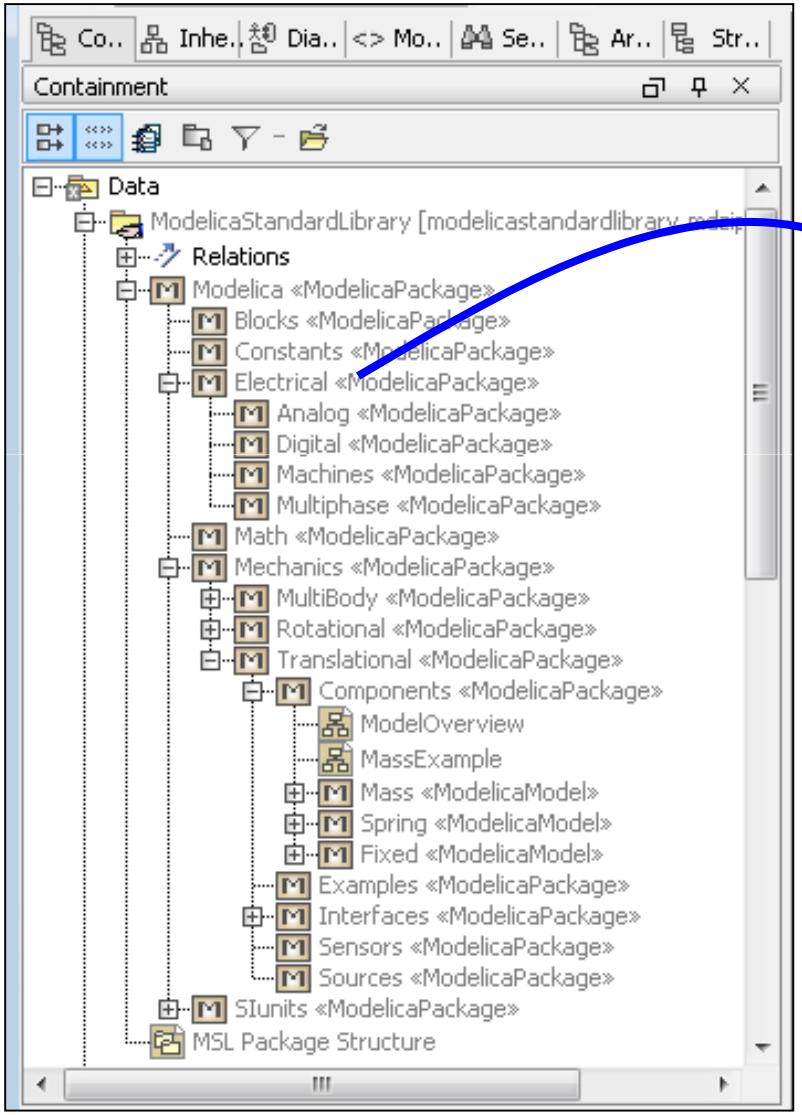


SysML-Modelica Robot Example: Analysis and Trade Study

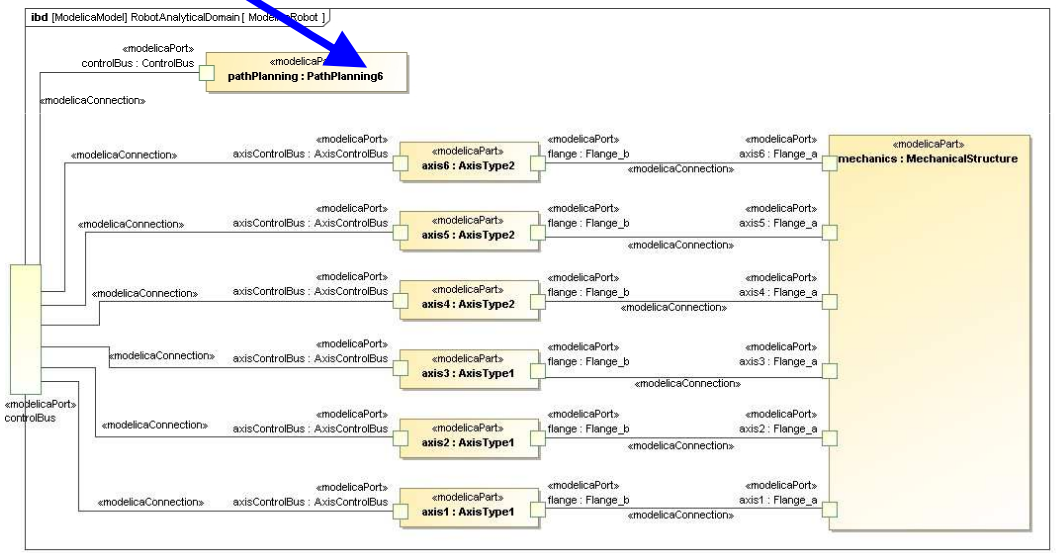


Analysis models depend on descriptive models

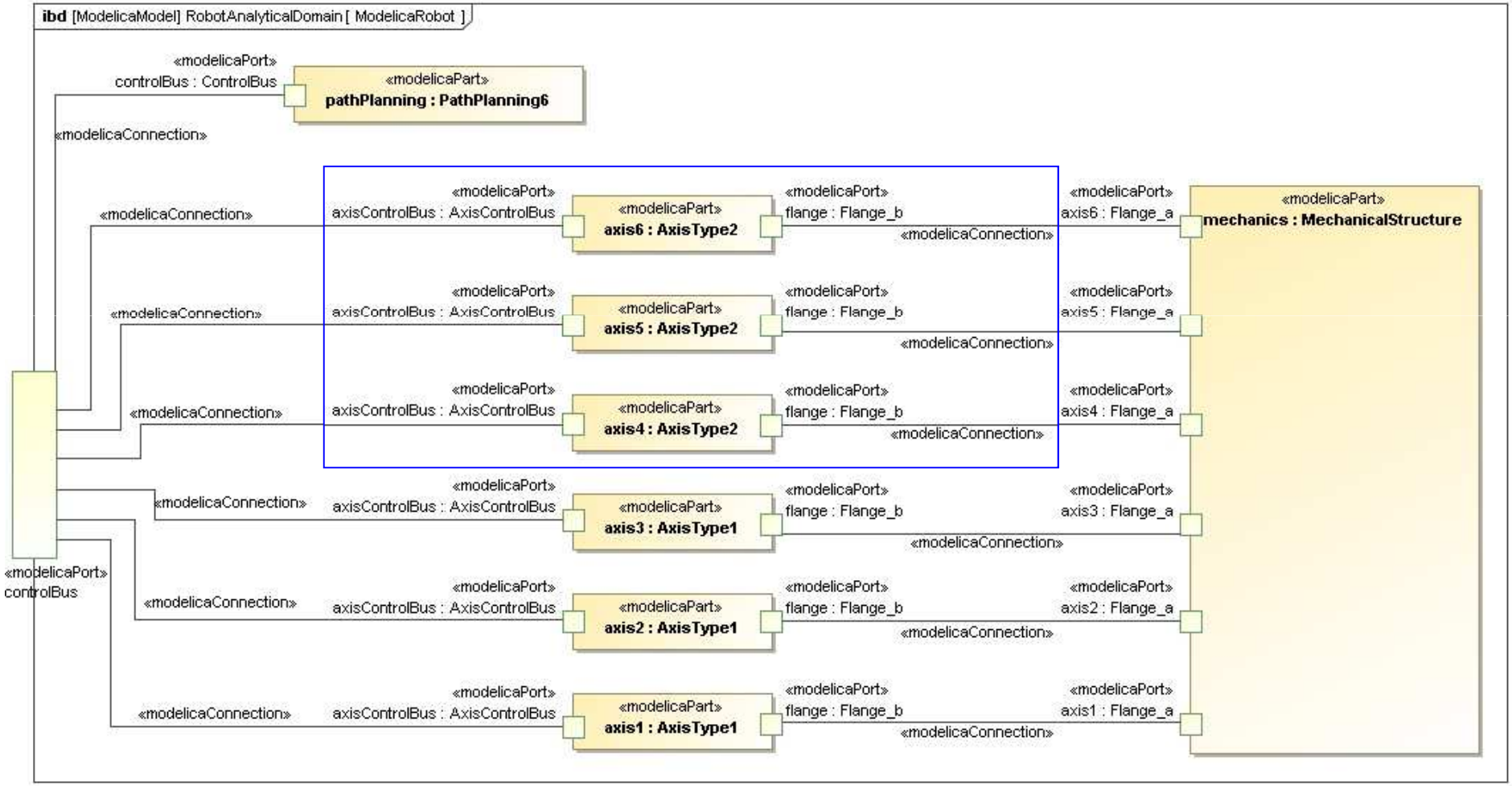
SysML4Modelica Analytical Model: Compose Model from Standard Library



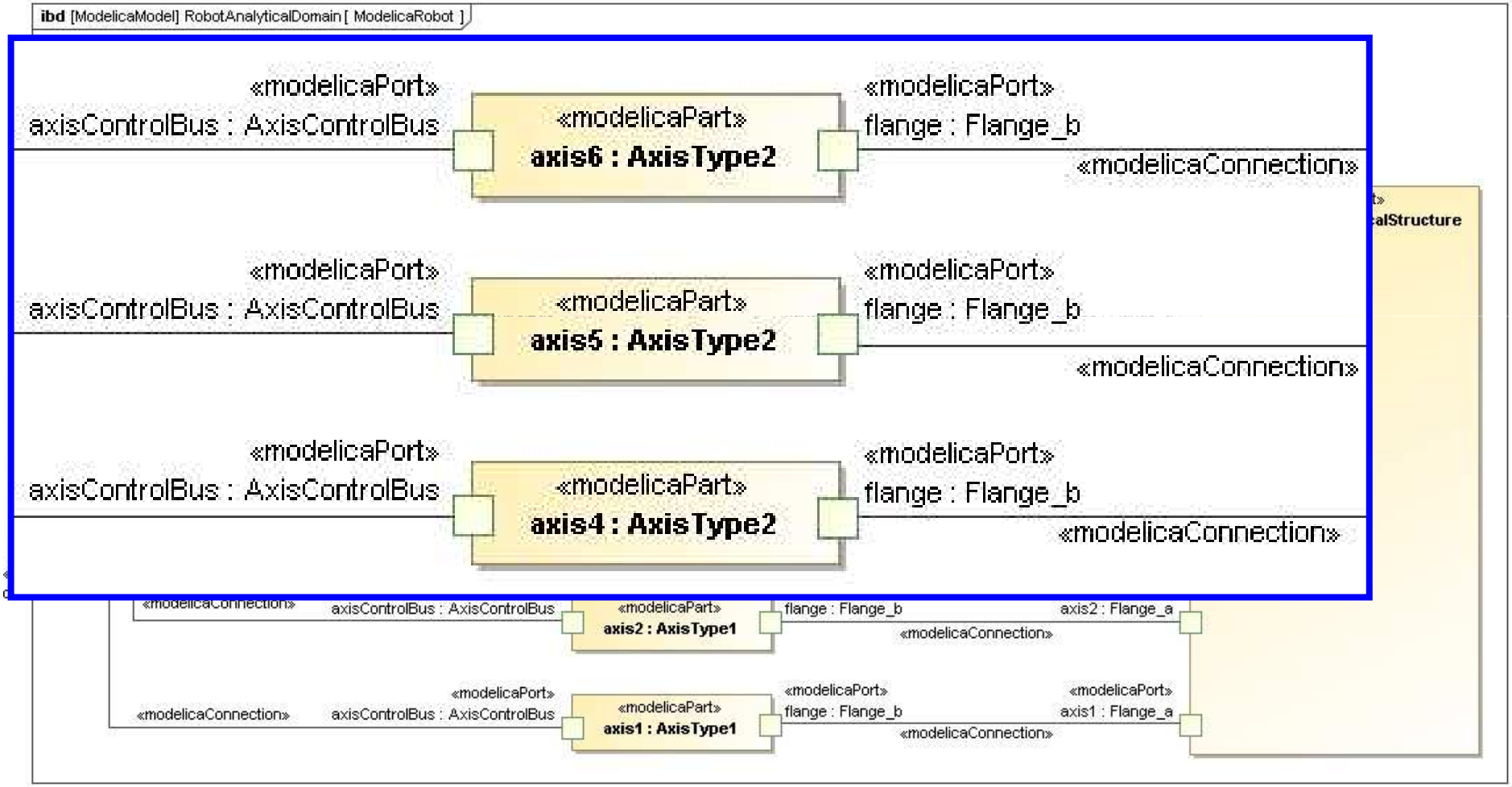
Drag and drop into IBD «ModelicaModel»



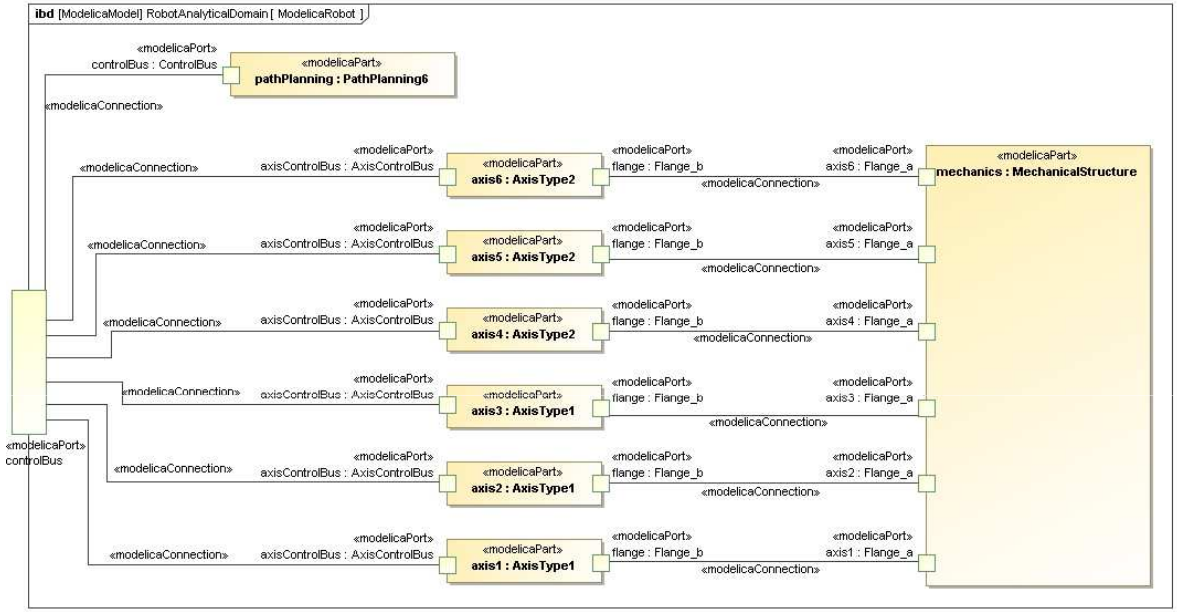
SysML4Modelica Analytical Model: Detailed IBD

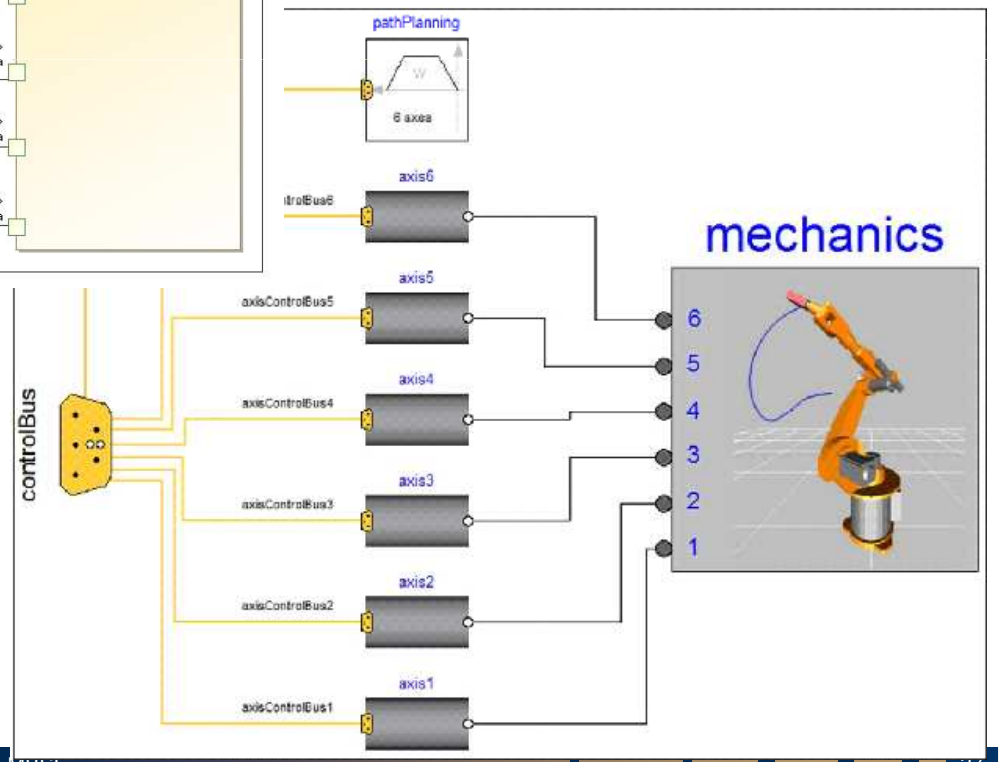


SysML4Modelica Analytical Model: Detailed IBD

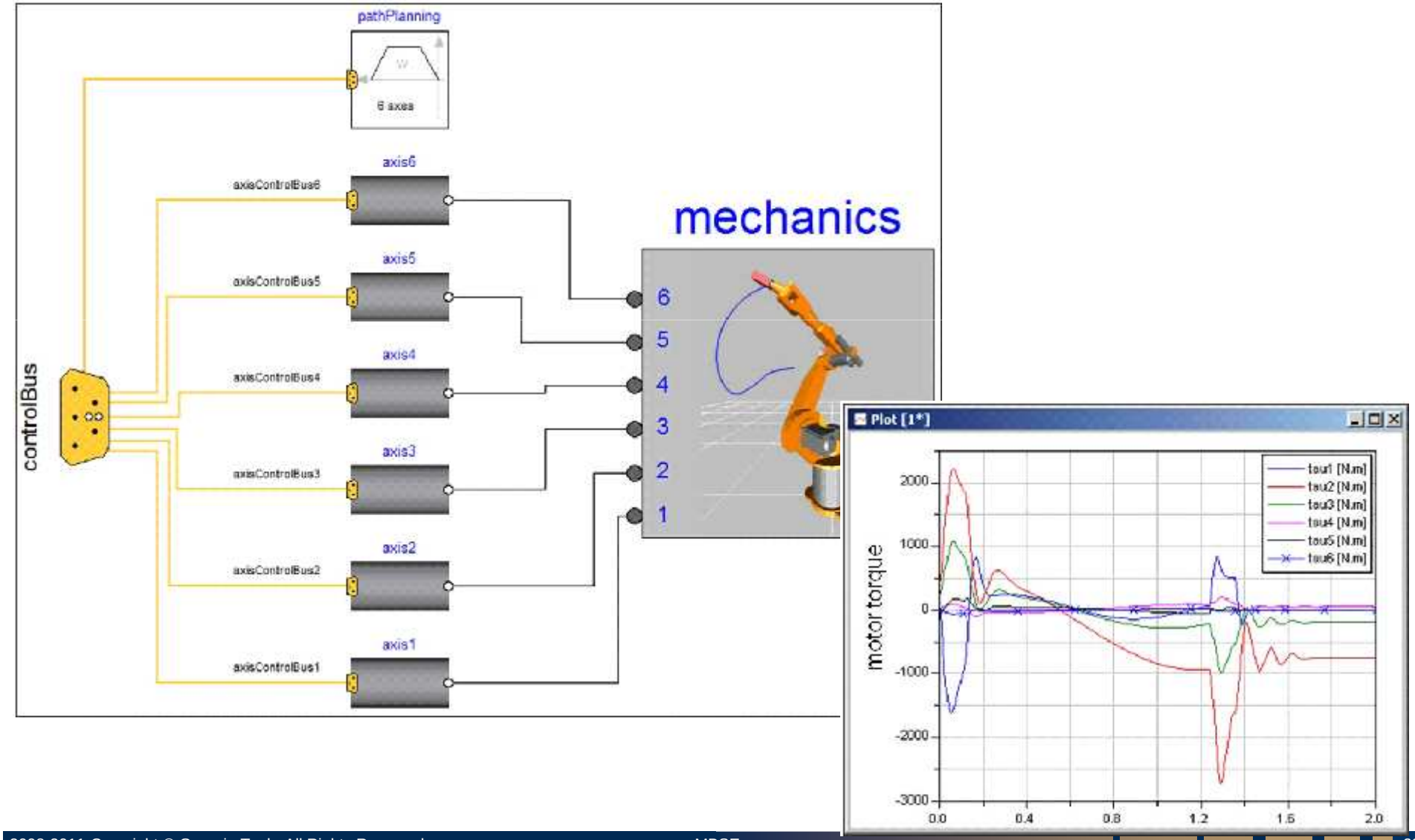


SysML4Modelica Analytical Model: Relation to Modelica Native Model

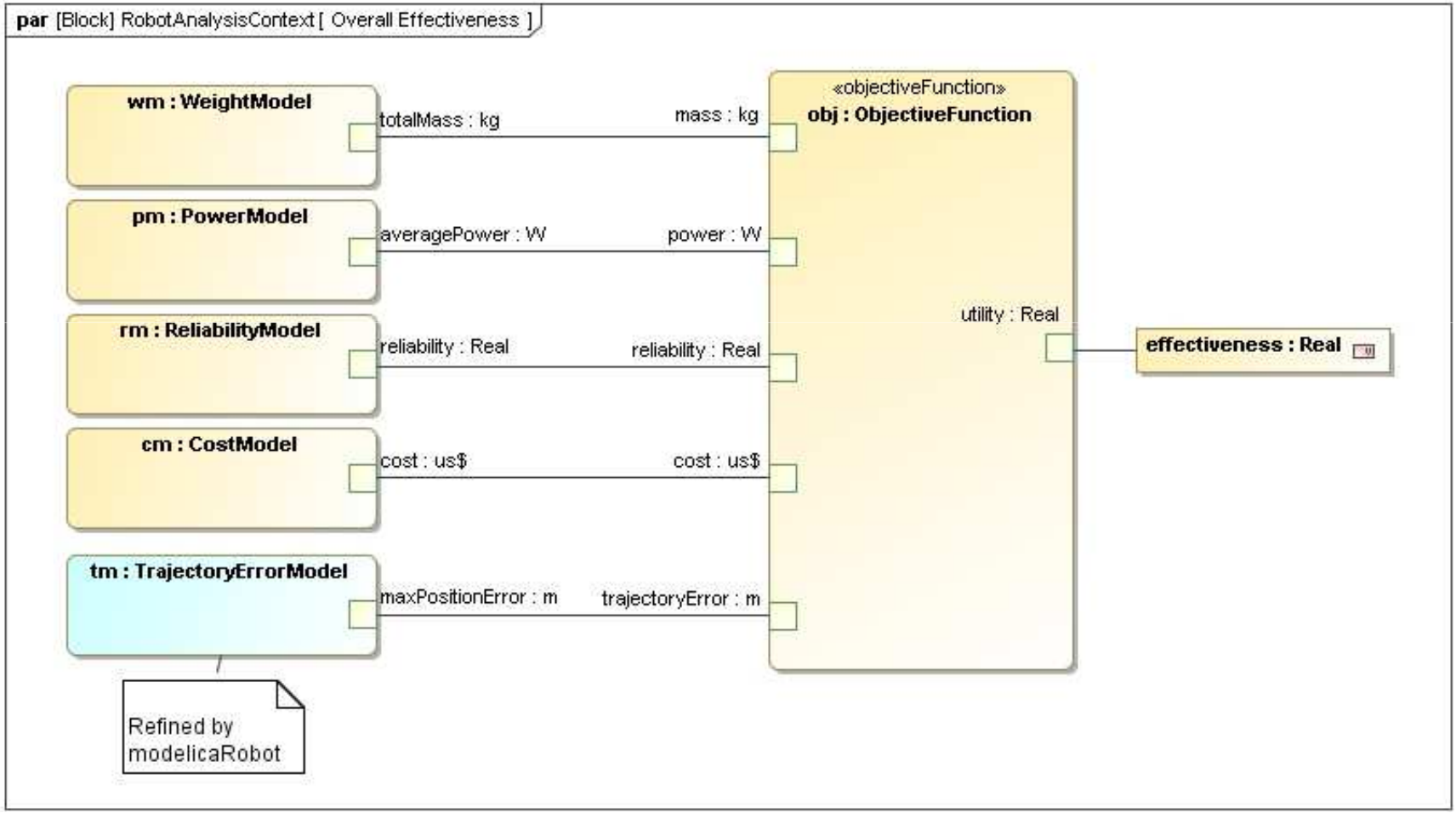




SysML-Modelica Robot Example: Modelica model with simulation results

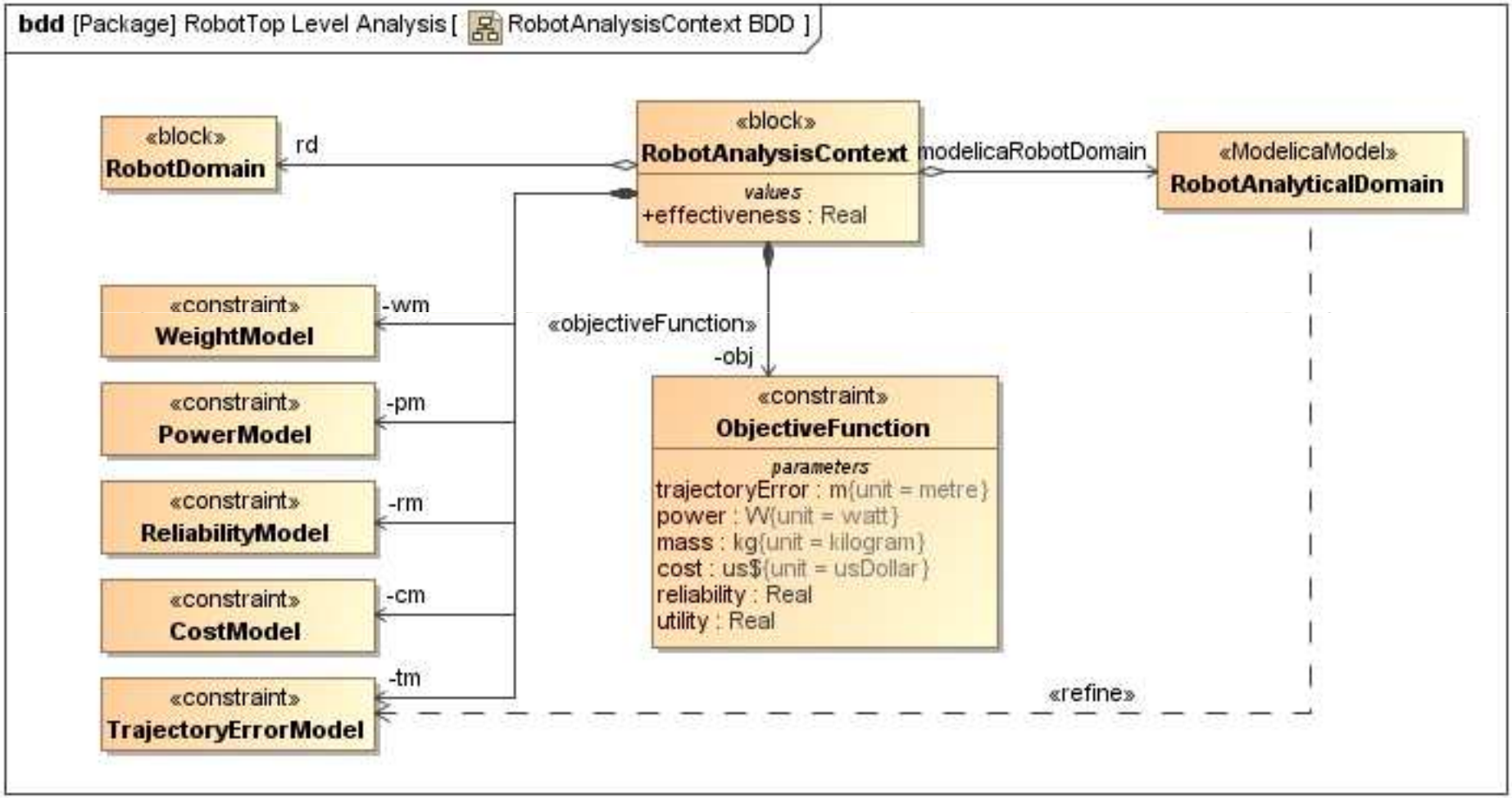


SysML-Modelica Robot Example: Analysis and Trade Study

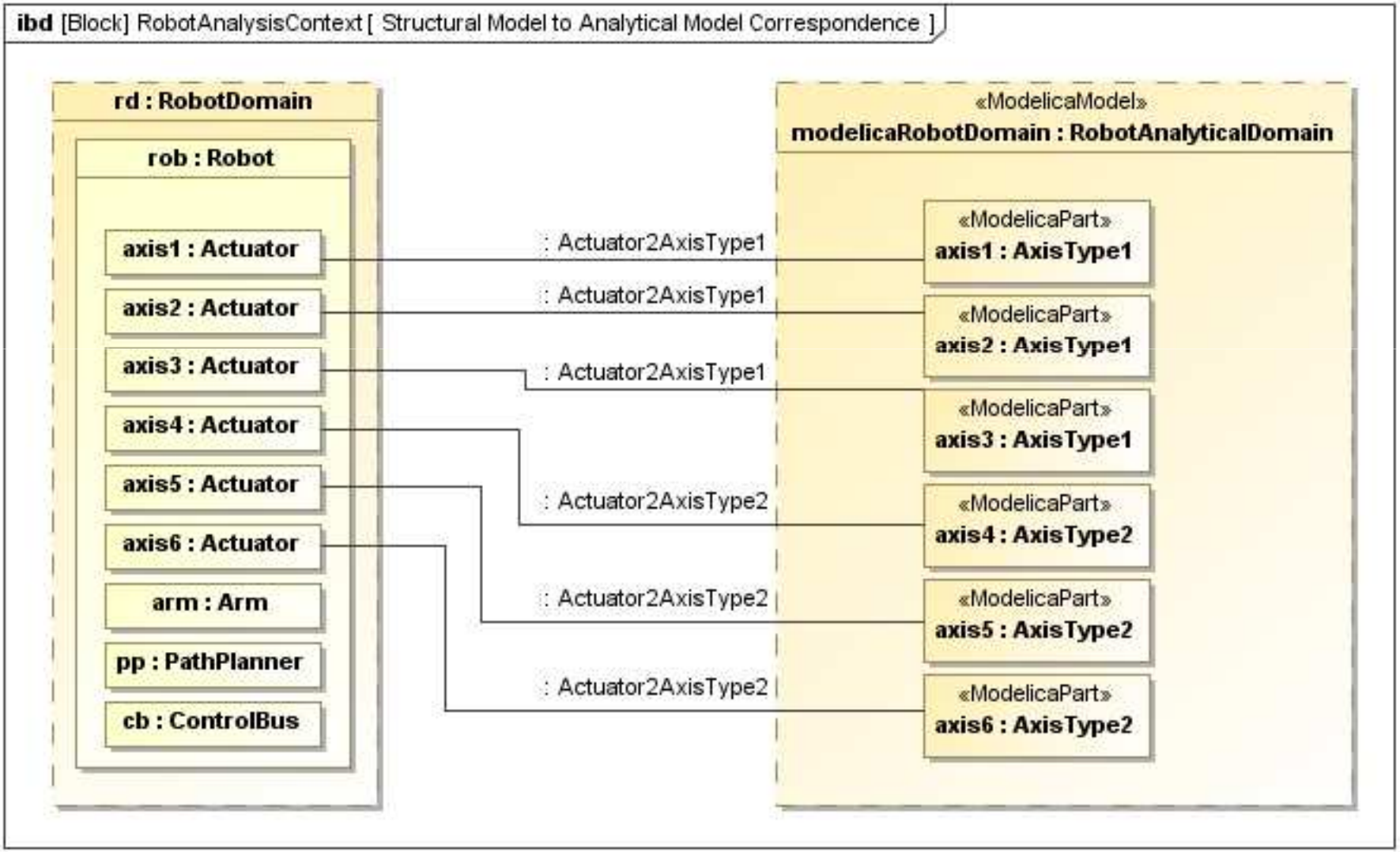


Analysis results are incorporated in Trade Study

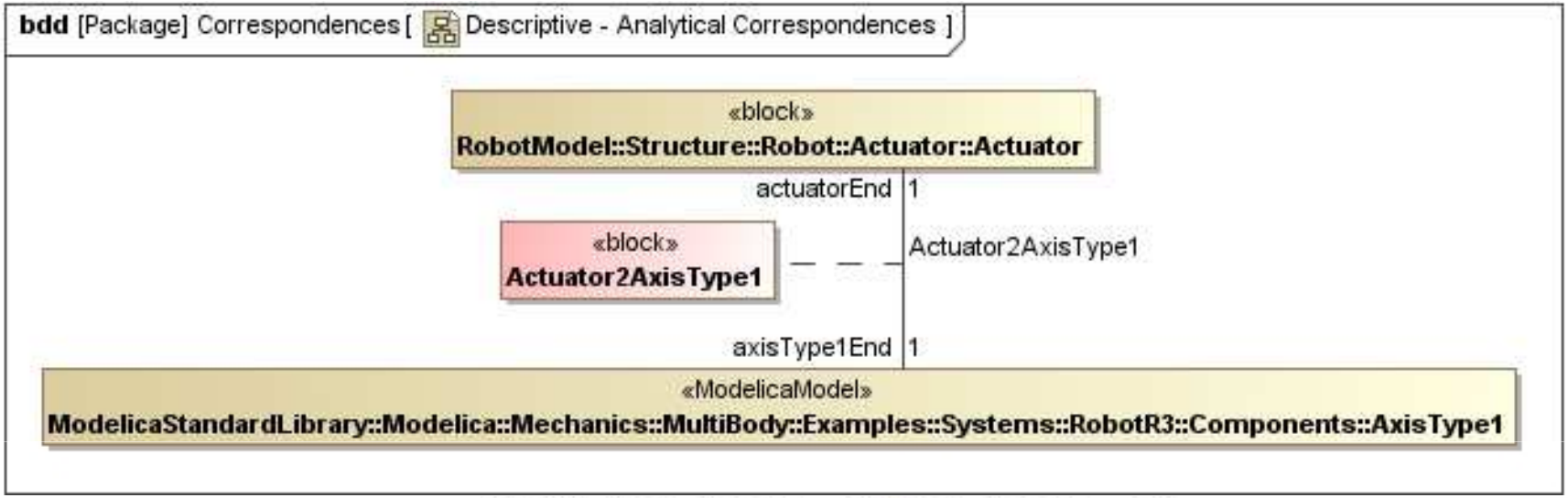
SysML-Modelica Robot Example: An Analysis Context

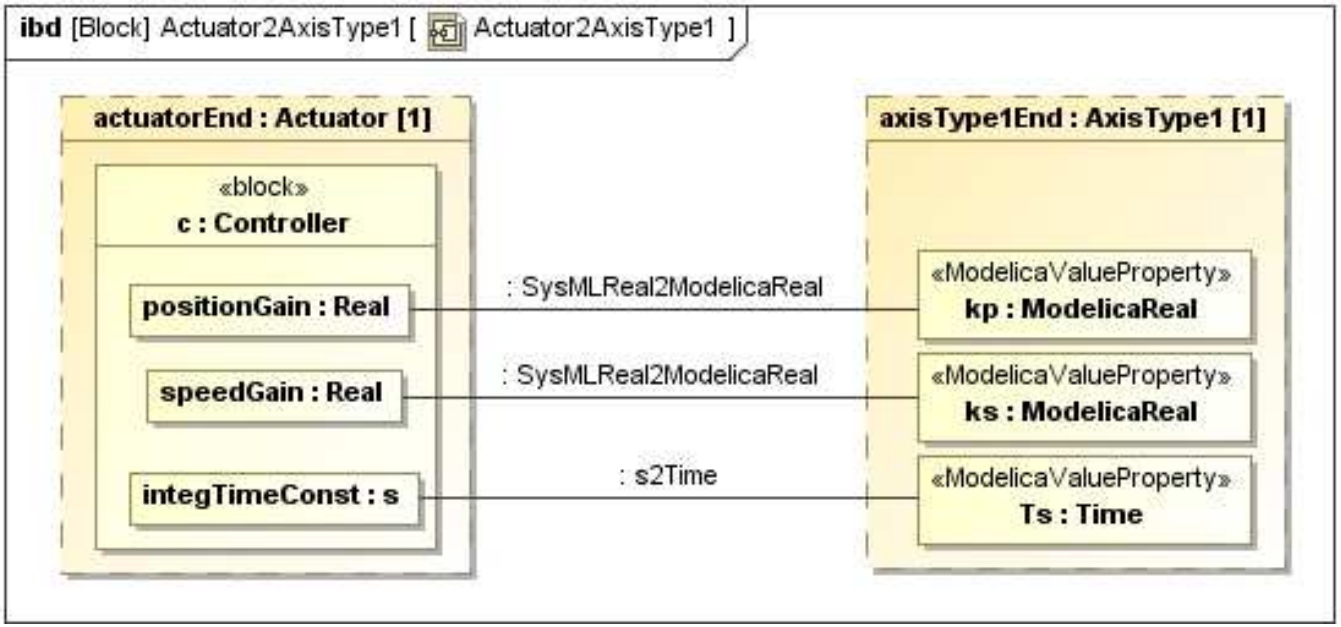


SysML-Modelica Robot Example: Dependencies between Models



SysML-Modelica Robot Example: Dependencies between Models





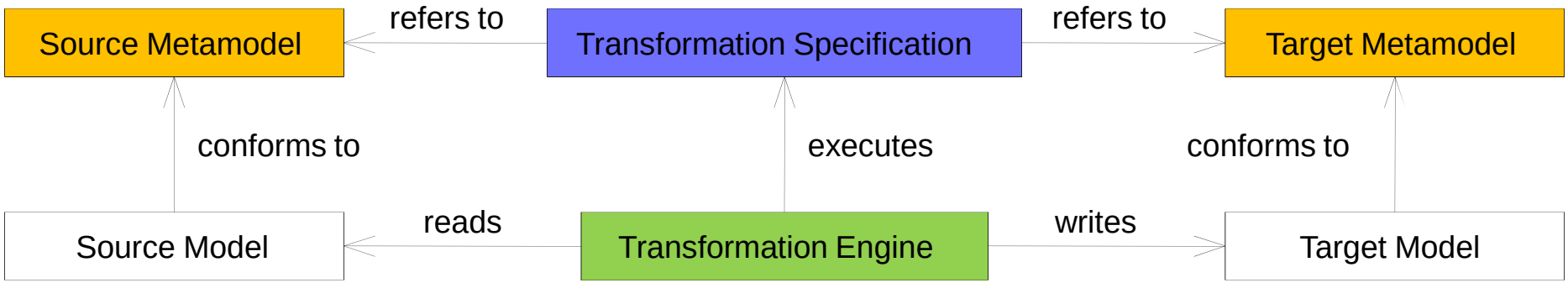
Presentation Overview

- What is SysML?
- What is Modelica?

- Motivating Example: Design & Analysis of Robot

- SysML-Modelica Transformation Specification
- Transformations in Model-Based Systems Modeling
- Summary

Model Transformation

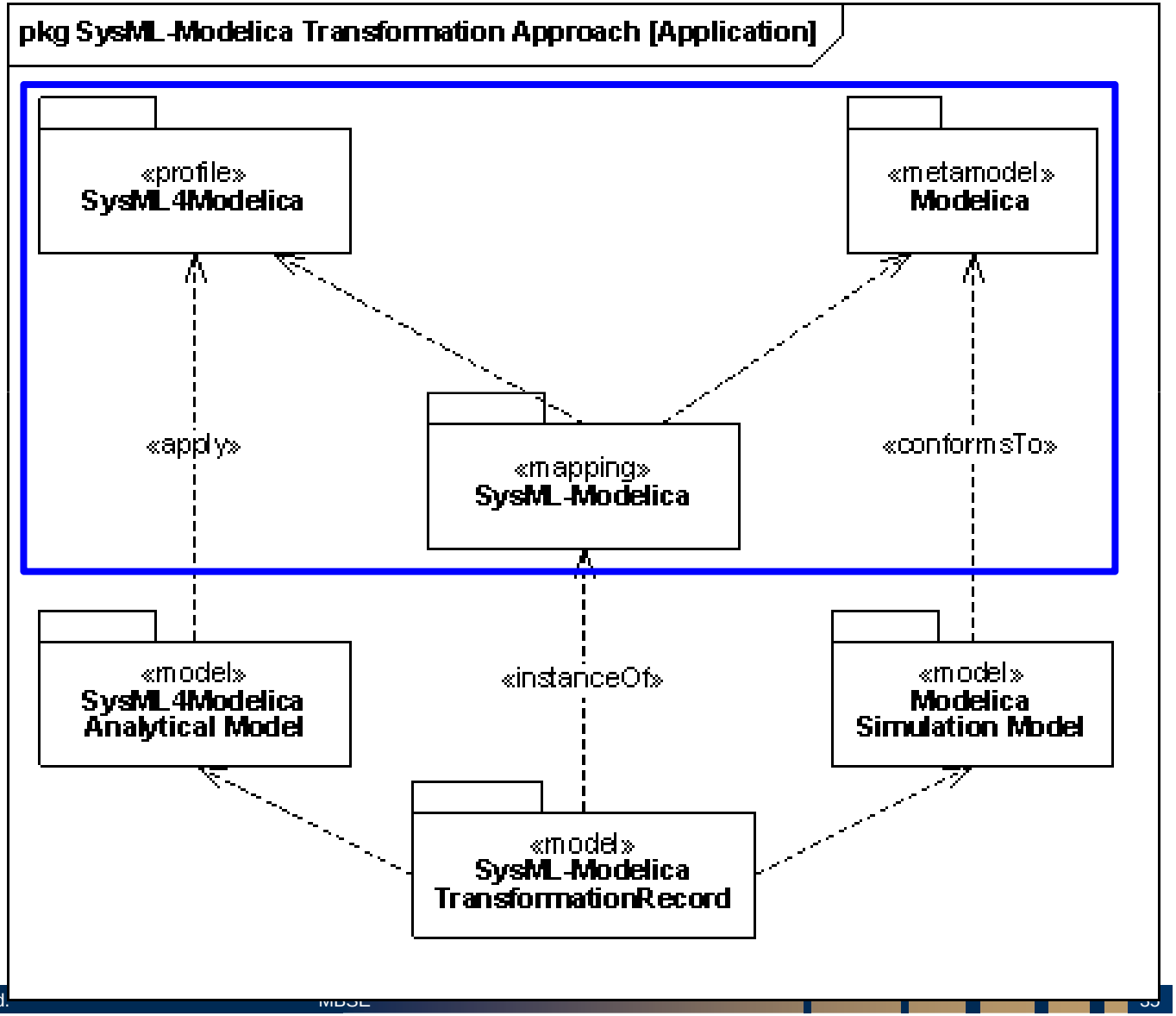


(Czarnecki, K., & Hellen, S., 2006)

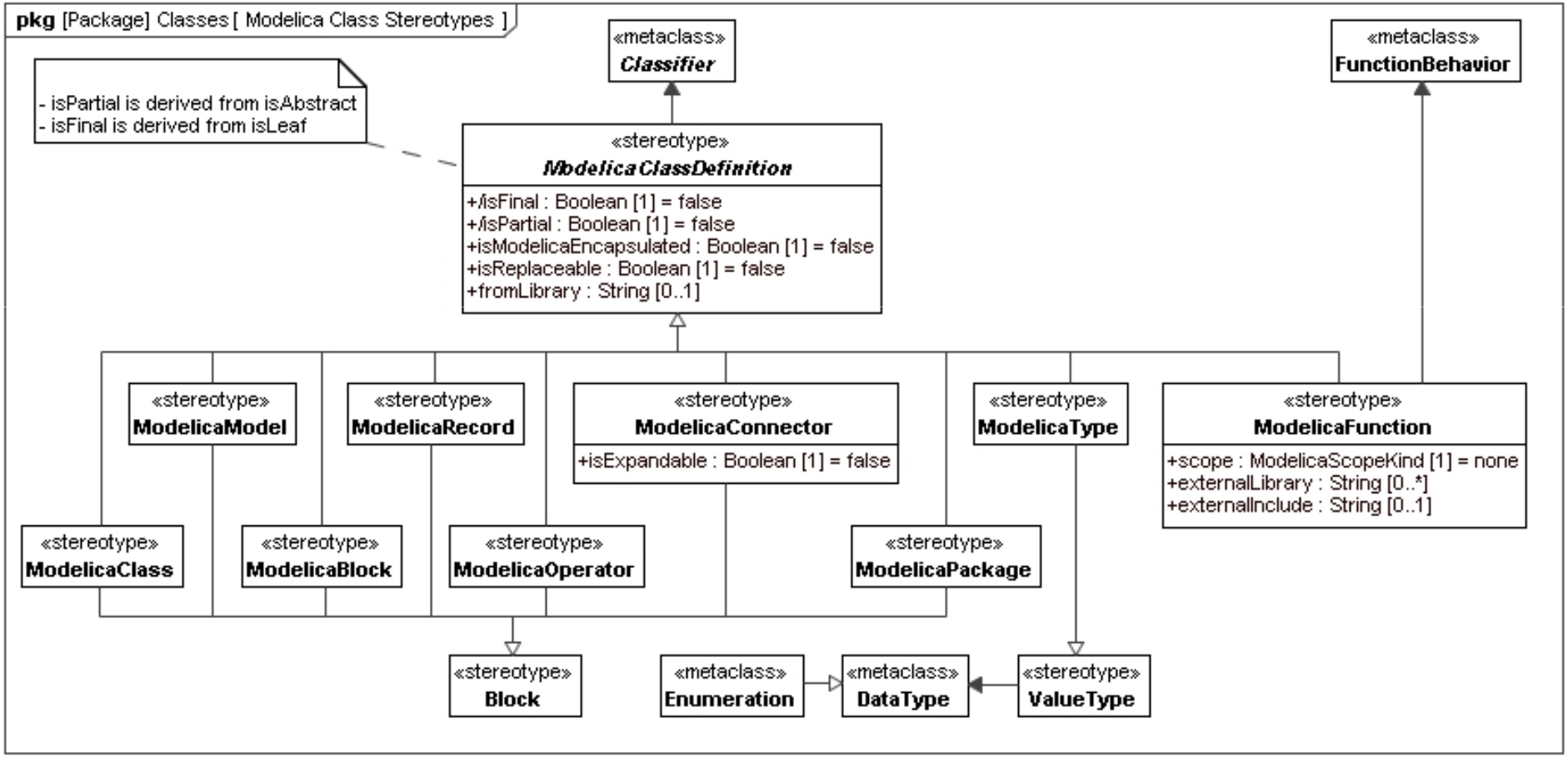
- Model Driven Architecture / Engineering
- Transformation Specification = Model
  → automated generation of transformation engine code
- Tools: MOFLON, QVTo, ATL, Kermeta, VIATRA2, …

SysML-Modelica Transformation Specification

SysML-Modelica Transformation follows the principles of Model-Driven Architecture (MDA)



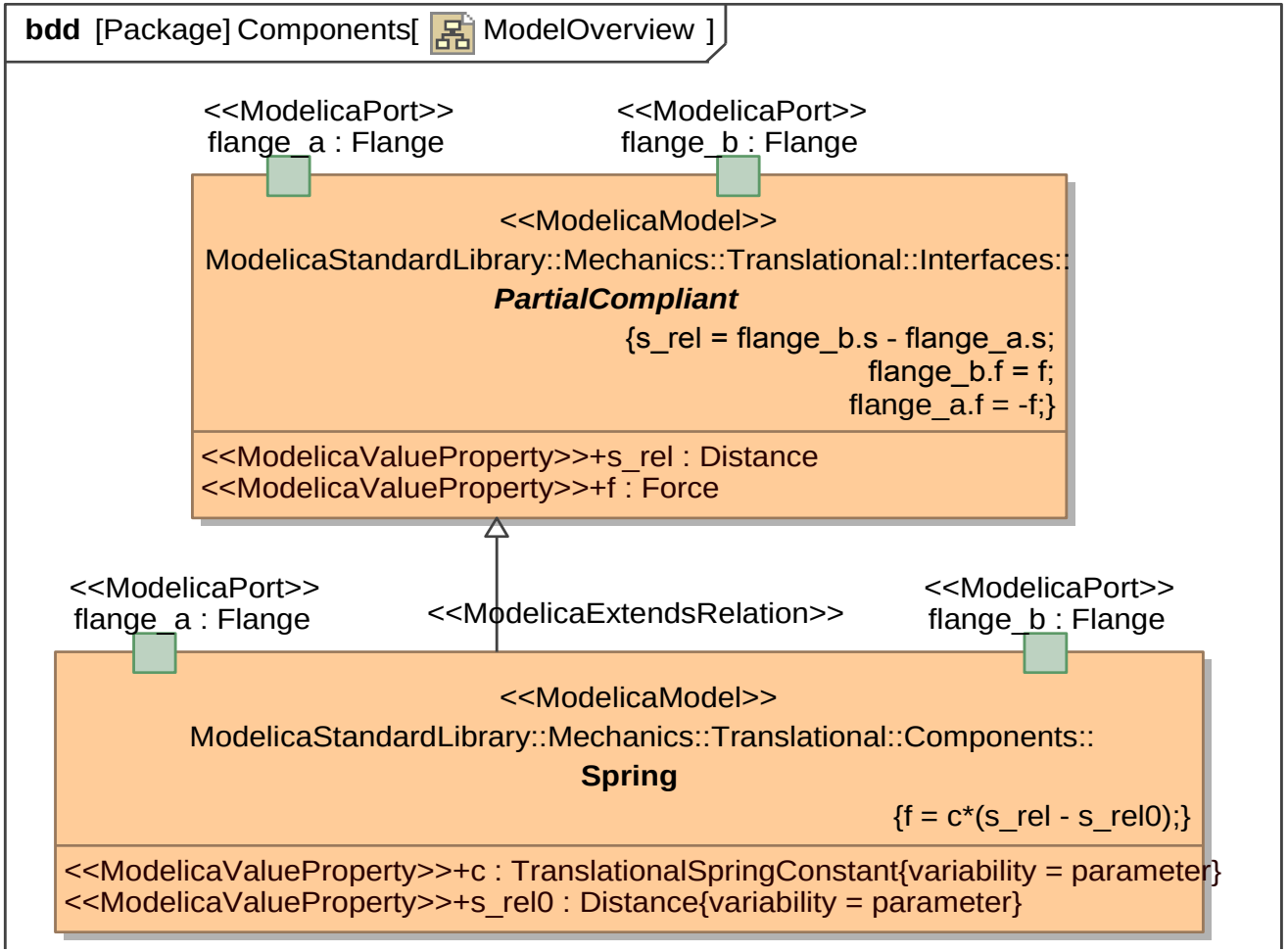
SysML4Modelica Profile



SysML4Modelica

Formal mapping

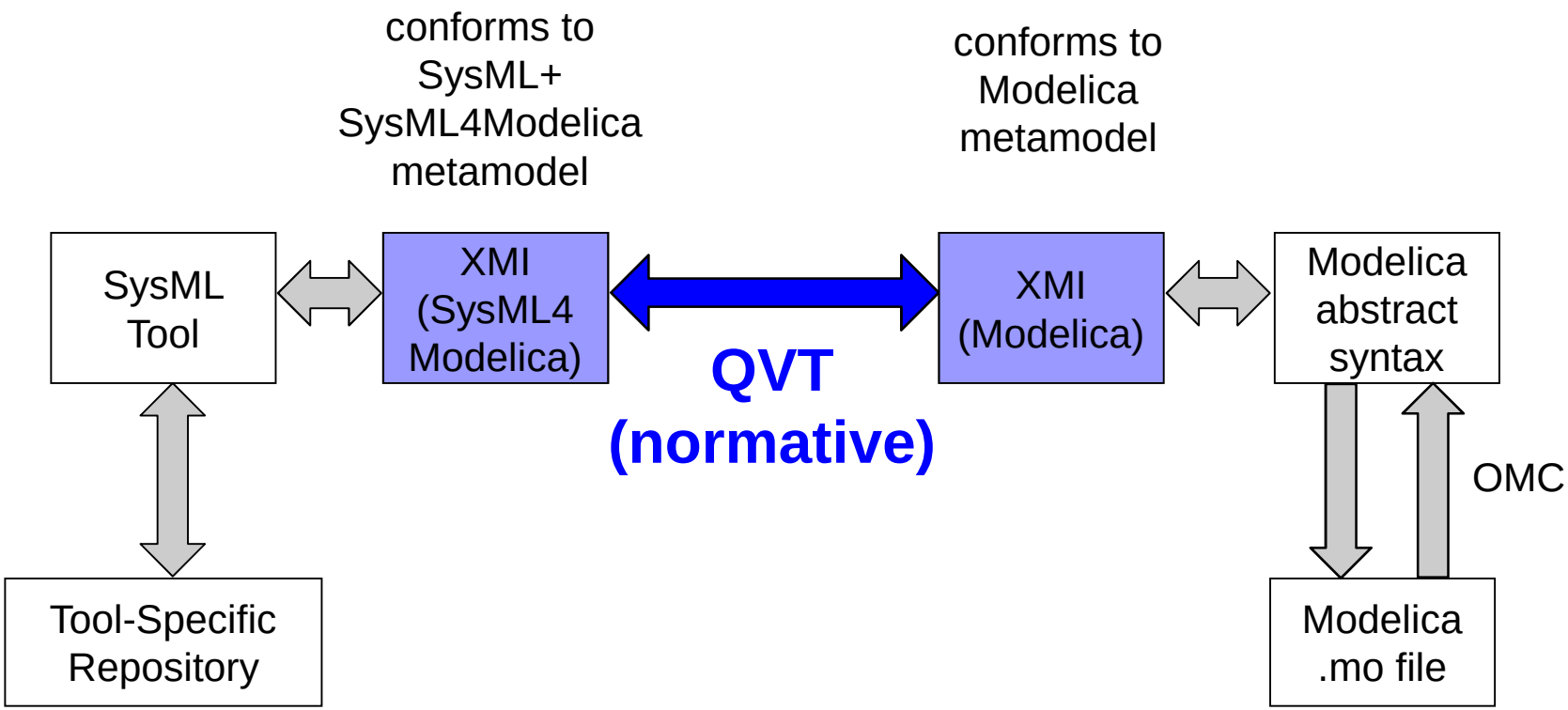
Modelica




```
model Spring "Linear 1D translational spring"
  extends Translational.Interfaces.PartialCompliant;
  parameter SI.TranslationalSpringConstant c(final min=0, start = 1)
    "spring constant ";
  parameter SI.Distance s_rel0=0 "unstretched spring length";

equation
  f = c*(s_rel - s_rel0);
end Spring;
```

Reference implementation: Based on OMG QVT



QVT = Query / View / Transformation

Presentation Overview

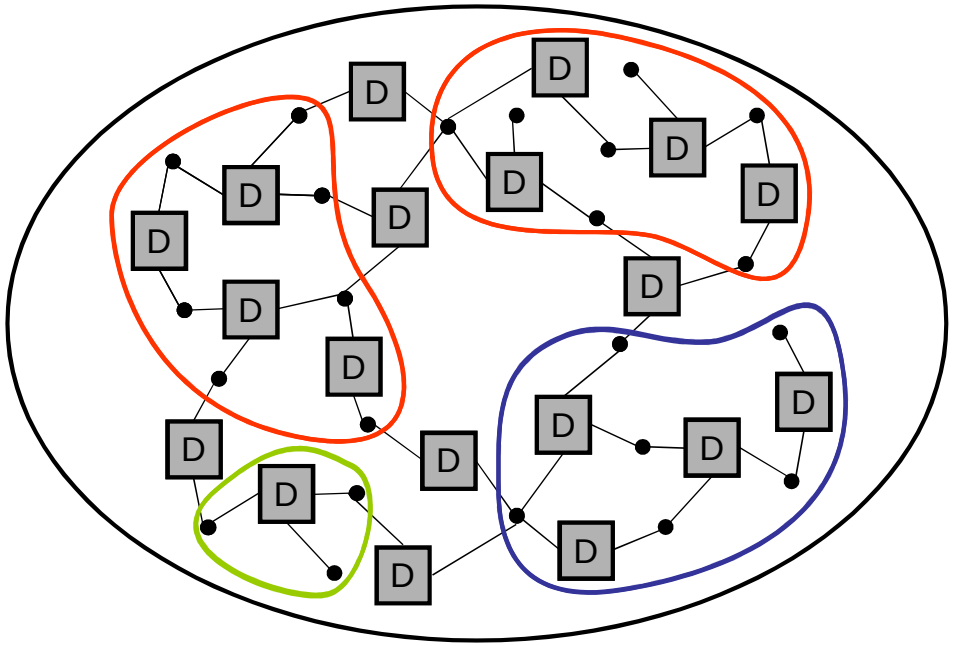
- What is SysML?
- What is Modelica?

- Motivating Example: Design & Analysis of Robot

- SysML-Modelica Transformation Specification
- Transformations in Model-Based Systems Modeling
- Summary

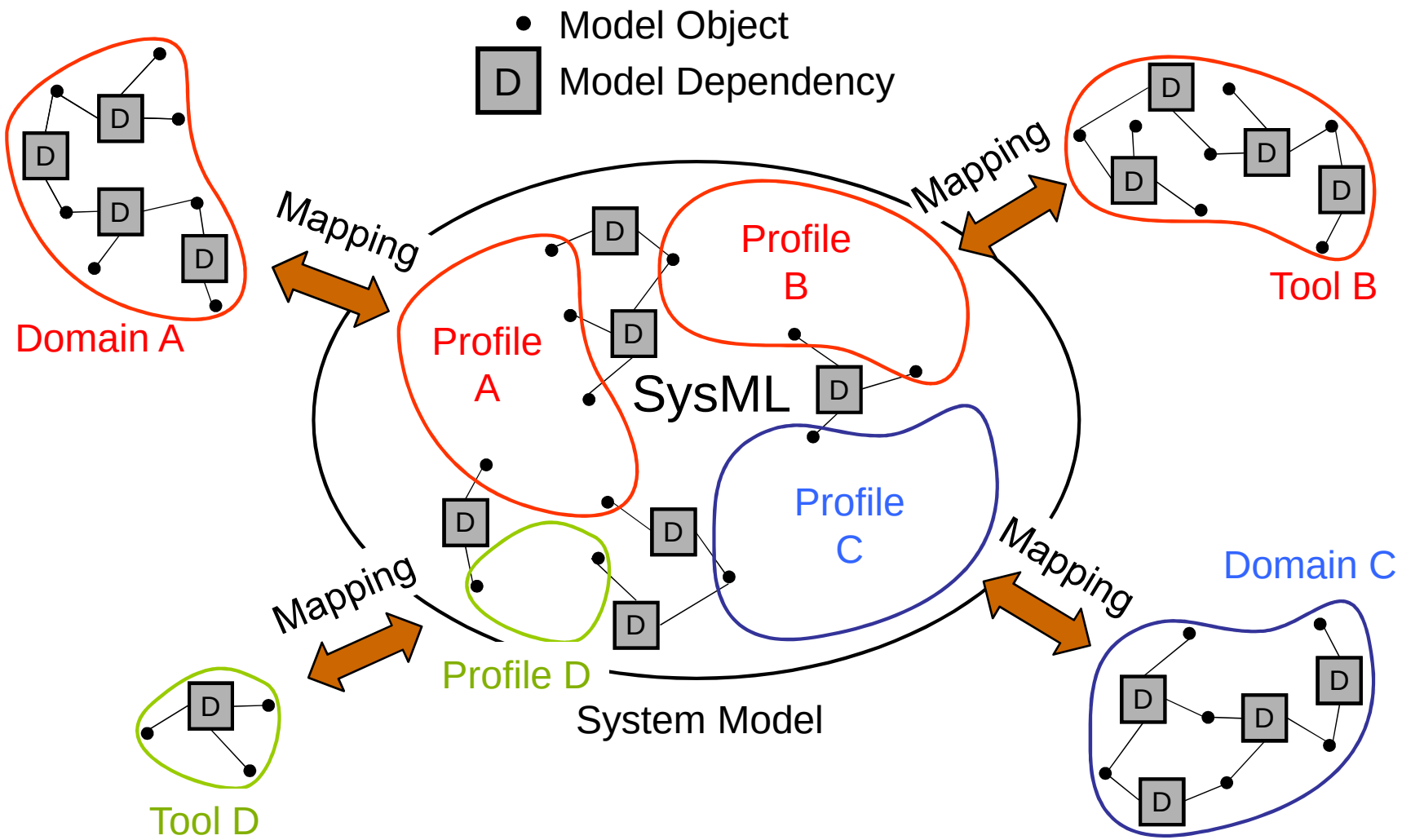
Model Transformations in MBSE

- Model Object
- D Model Dependency

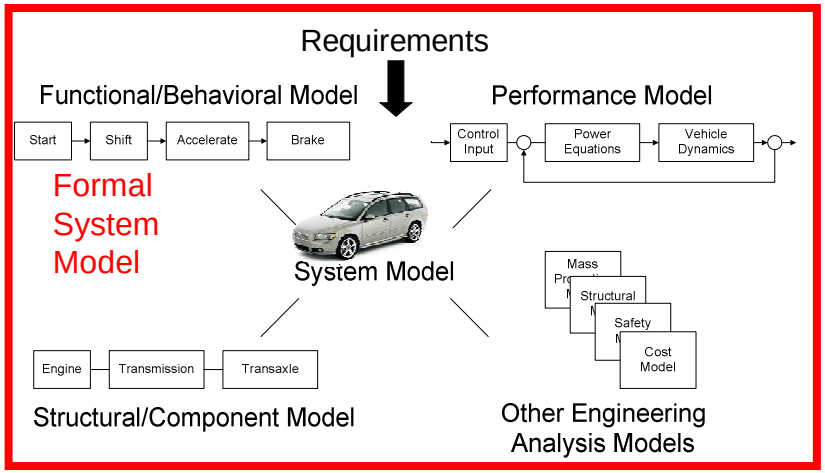


System Model

Model Transformations in MBSE: Federated Models



Model Transformations in MBSE



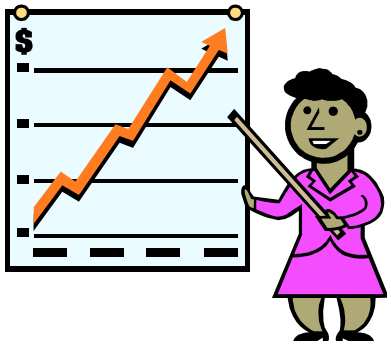
Transformation

Stage-Gate Documents

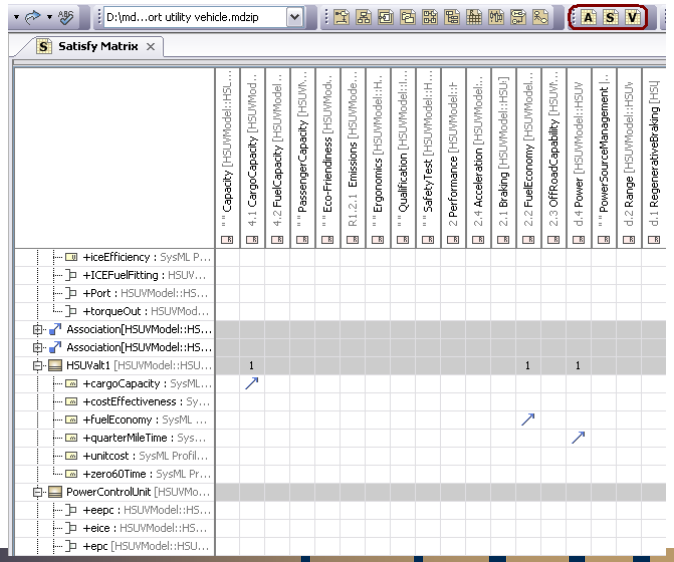
Transformation

Transformation

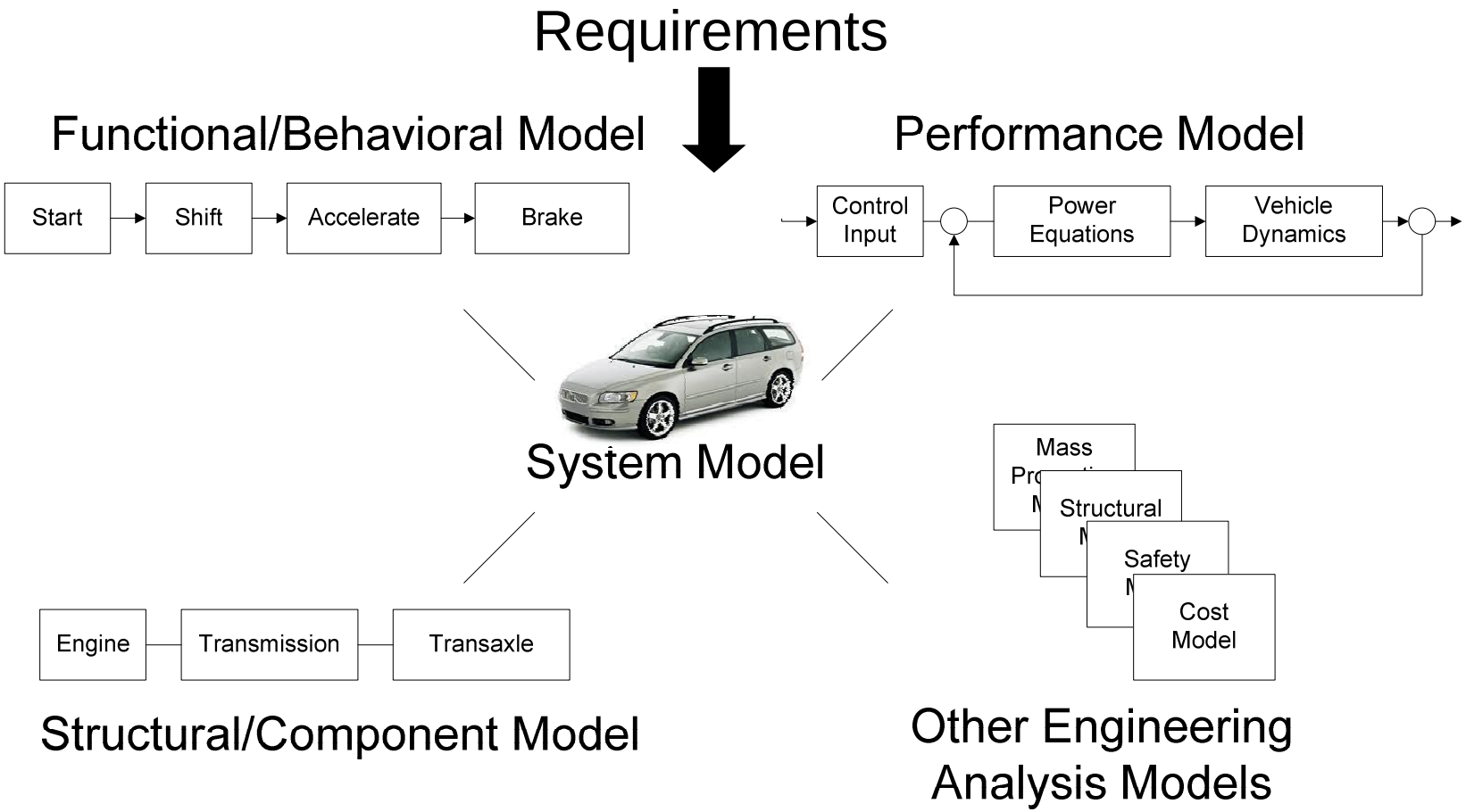
Project Management Metrics



Verification & Consistency Management



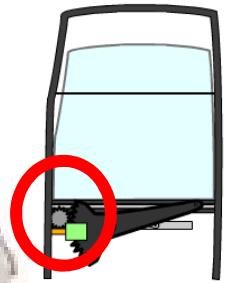
Model Transformations in MBSE: Reusable Models



Creating Models is Expensive and Time-Consuming

Model Transformations in MBSE: Applying Reuse Patterns

- Physical components are reused
- Portions of the system model repeat
- Patterns for instantiating these portions



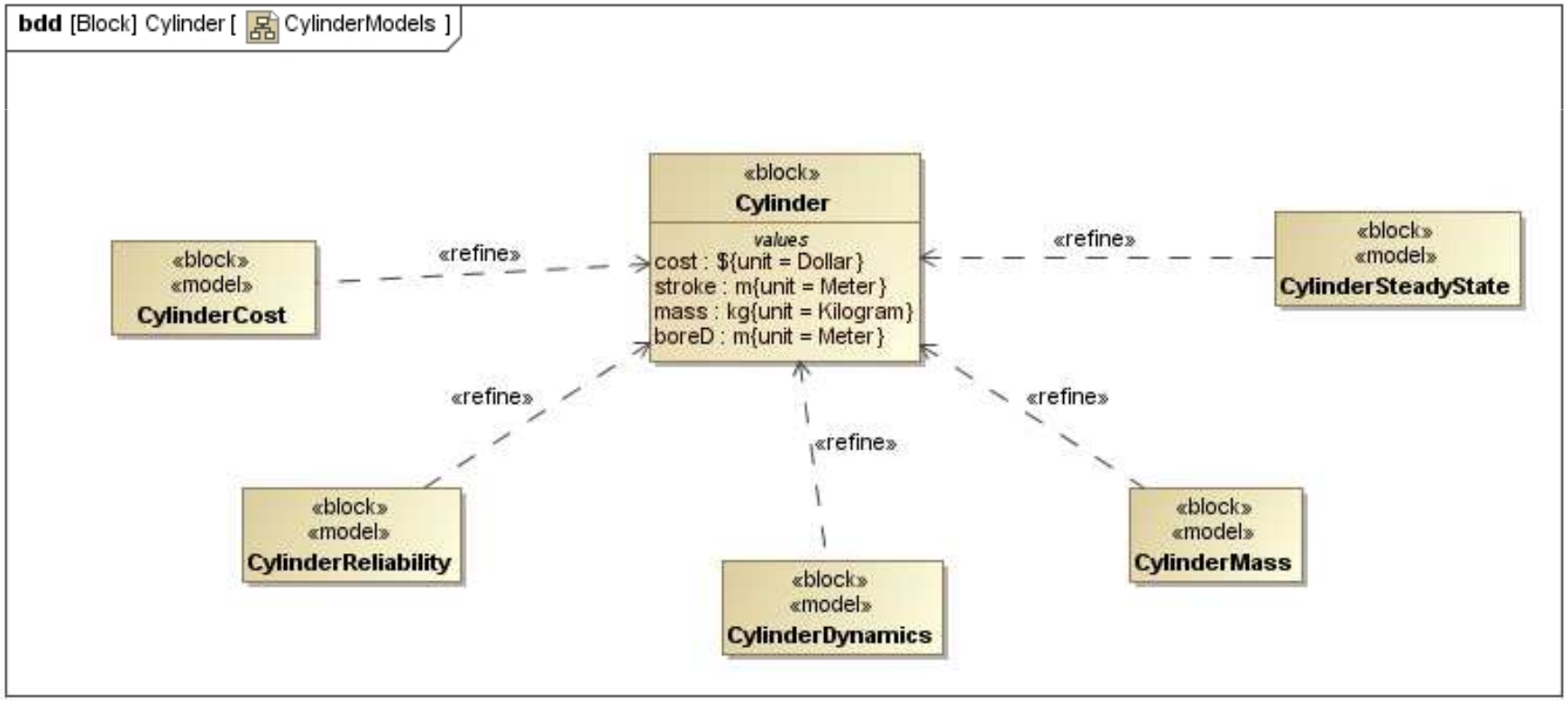
Electric Motors

- Component models → Domain specific model libraries
- Application of pattern = model transformations

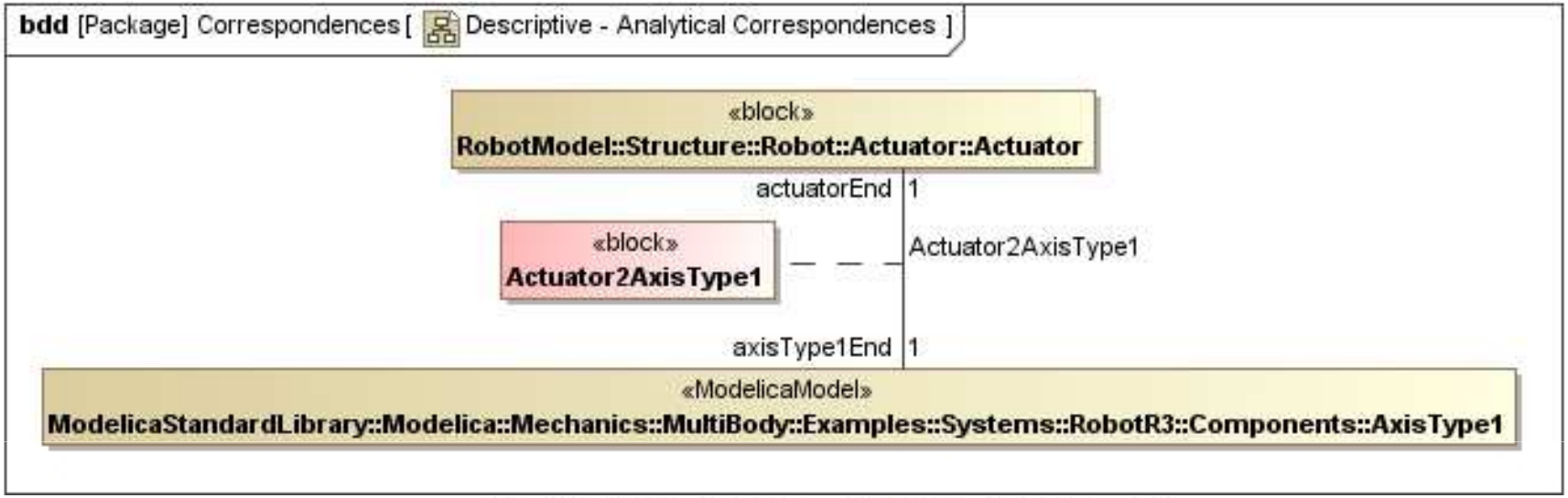
Model Transformations in MBSE: Applying Reuse Patterns

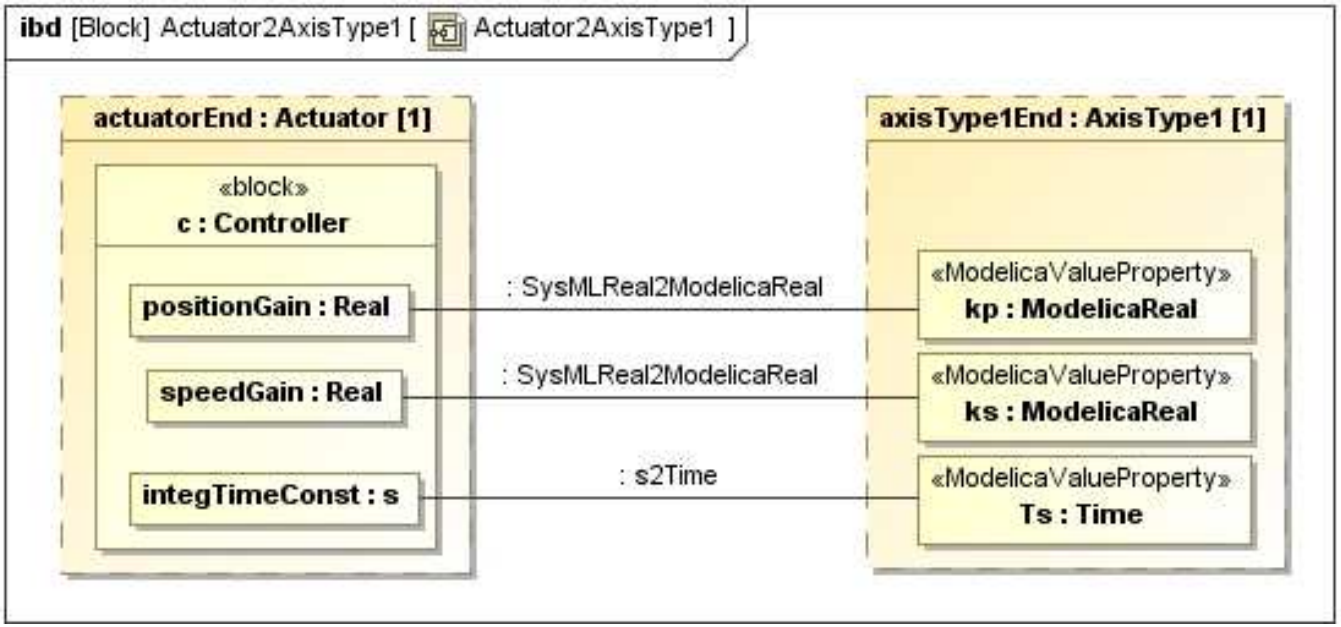
- When cylinder is used, other corresponding models are often used also

→ Capture the reuse pattern

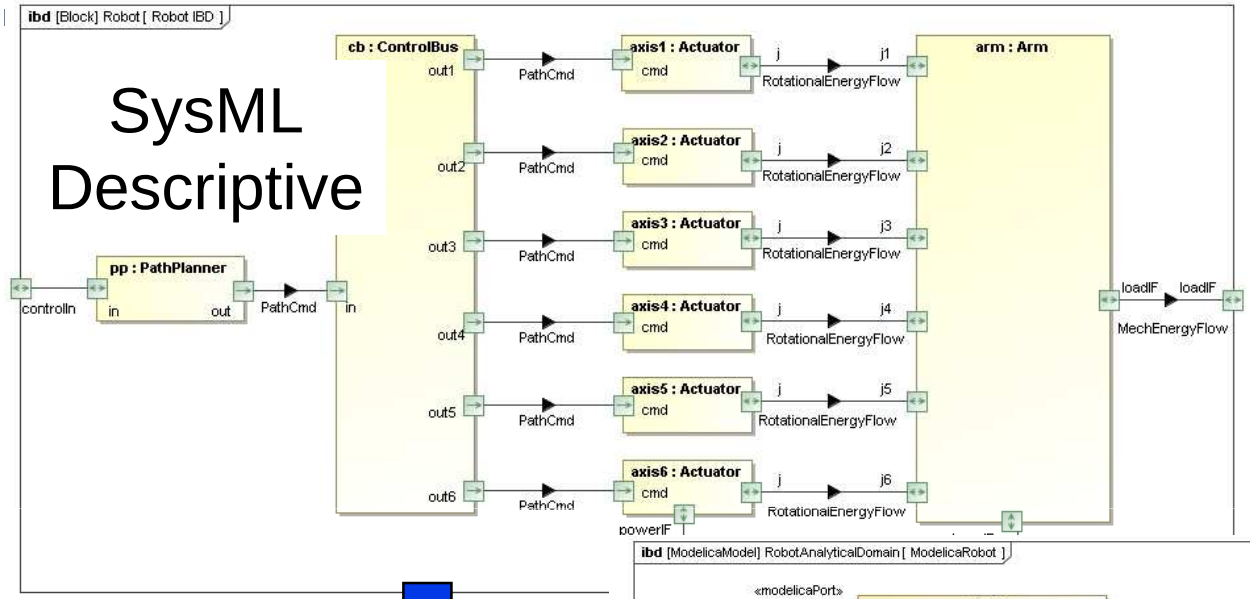


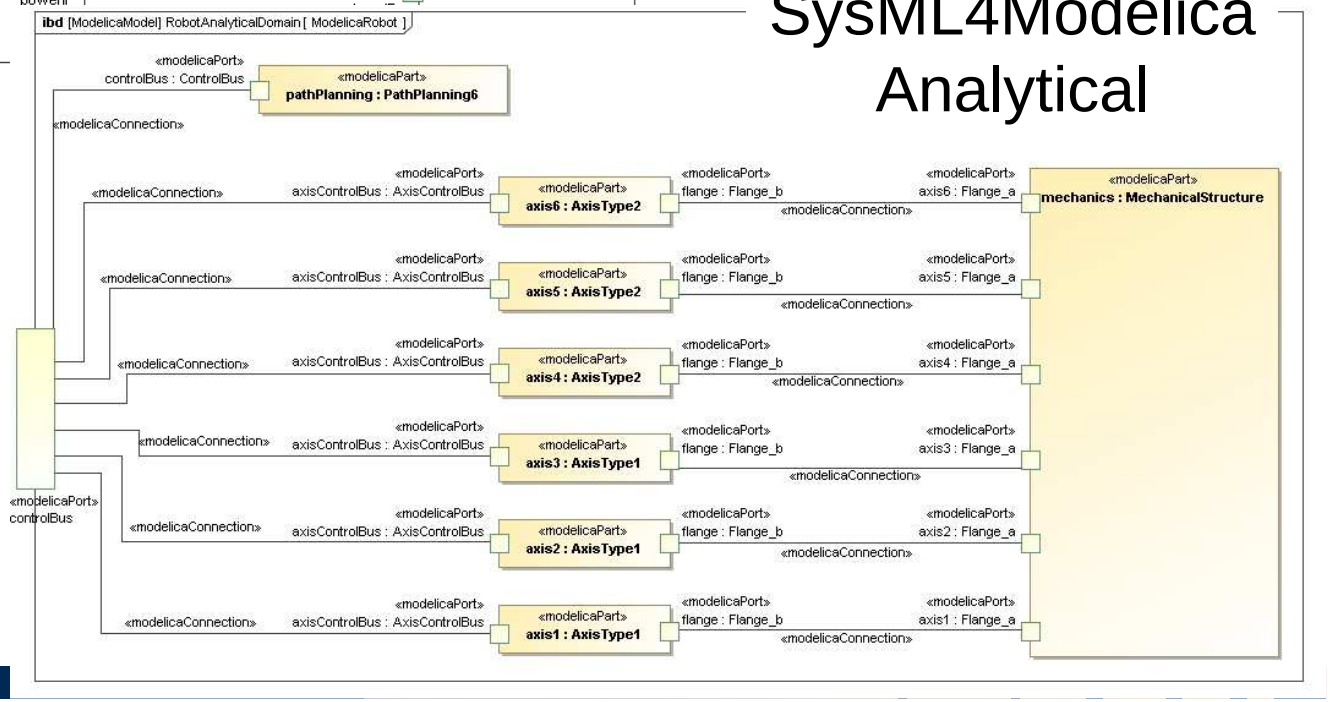
Model Transformations in MBSE: Applying Reuse Patterns

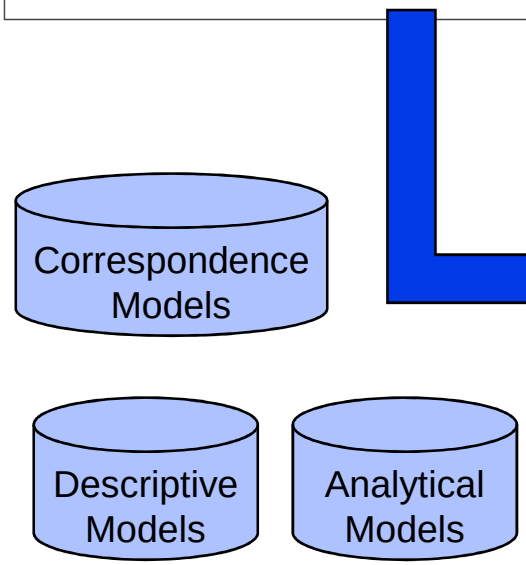




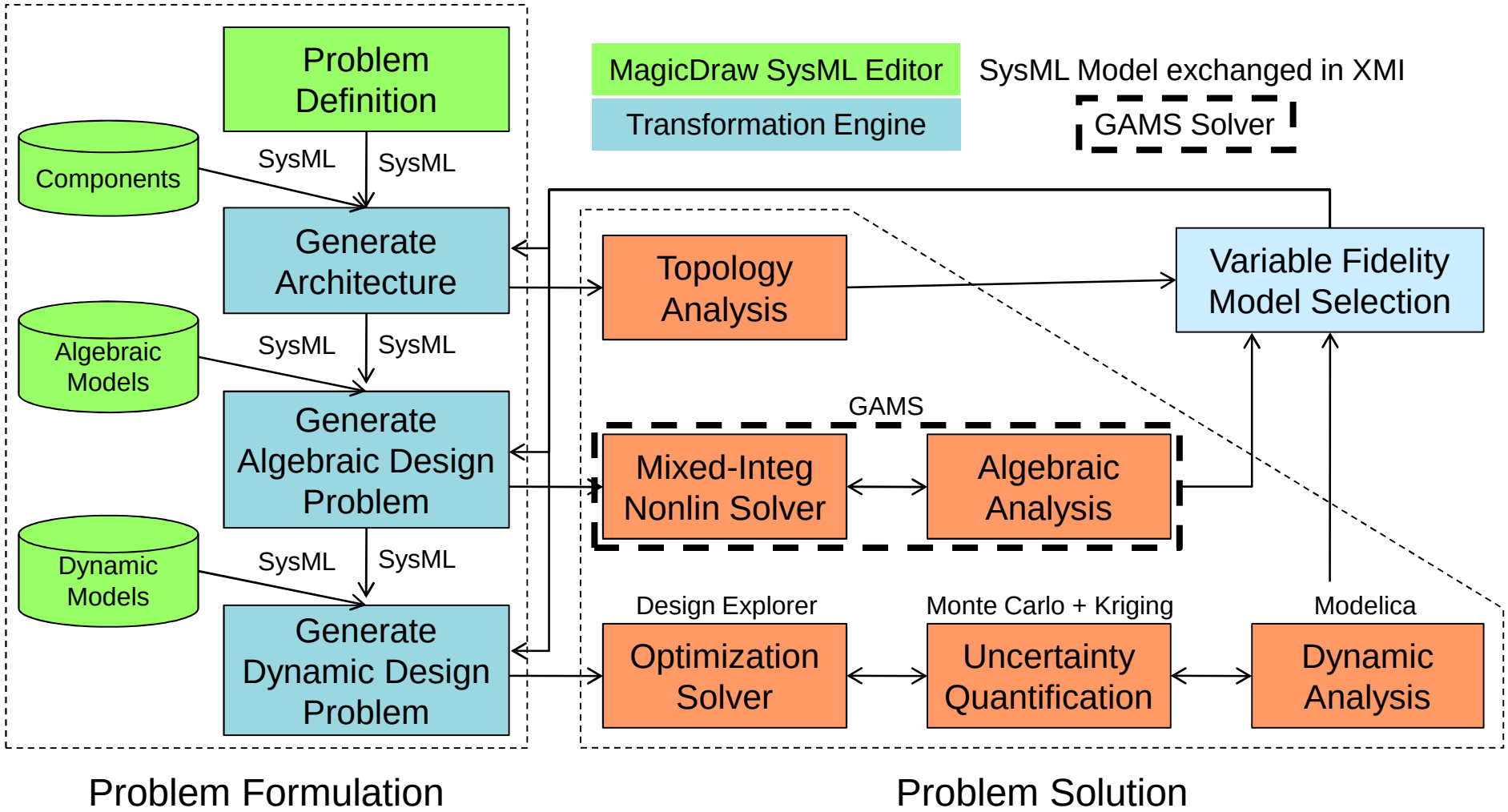
Model Transformations in MBSE: Descriptive to Analytical Transformation







Model Transformations in MBSE: Architecture Exploration



Summary

- Objective:
  - Leverage the strengths of both SysML and Modelica by integrating them to create a more expressive and formal MBSE language.

Descriptive Modeling in SysML

+

Formal Equation-Based Modeling for Analyses and Trade Studies in Modelica

http://www.omg.org/spec/SyM/