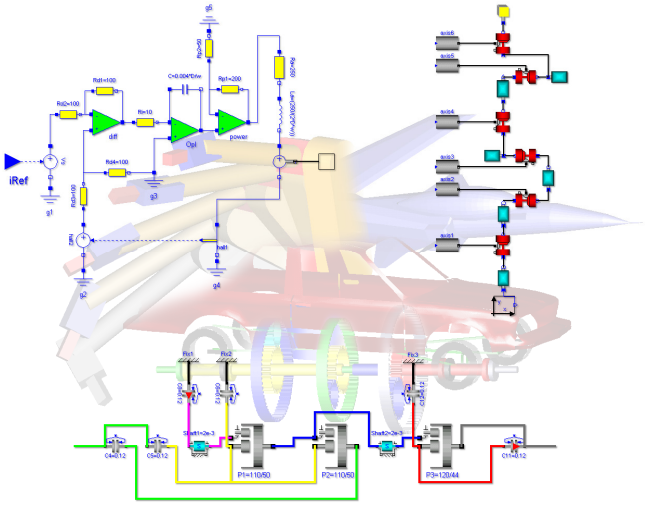
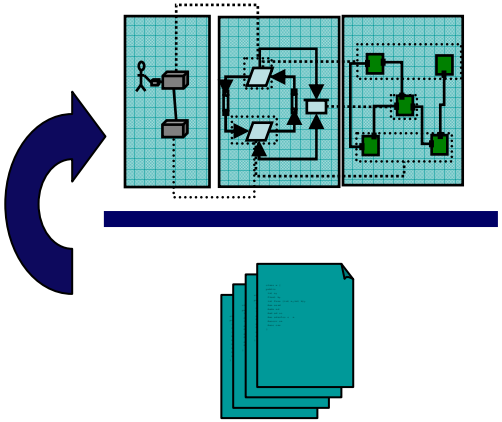
The Impreciseness of UML and Implications for ModelicaML

EOOLT, Paphos
2008-07-08

Jörn Guy Süss, Peter Fritzson, Adrian Pop

Presentation by Peter Fritzson





Outline

- Whole Product Modelling (Software/Hardware) implies Tool Integration
- Why UML is Bad (for Integration)
- Meta-Models and Frameworks
- An Example of Whole Product Modelling

Notations for Whole-Product Modeling

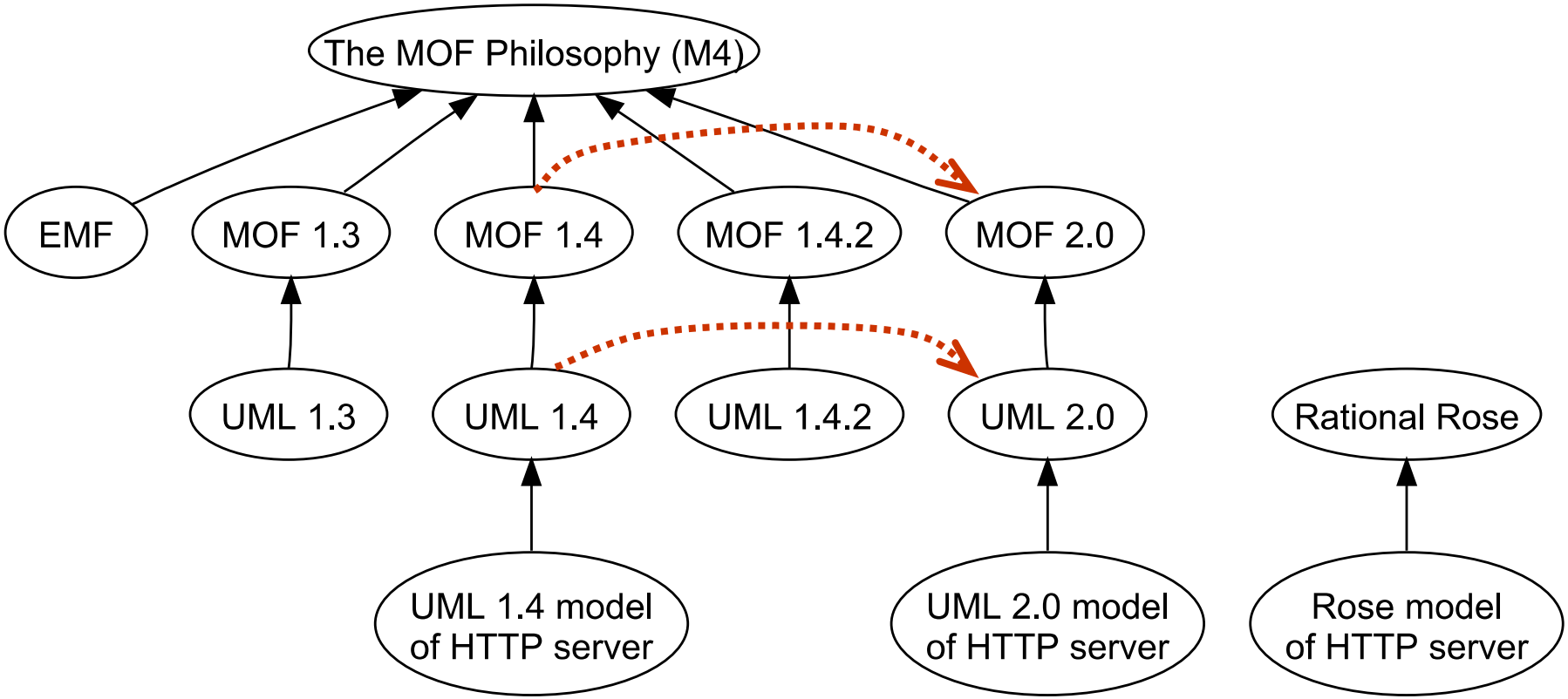
- UML is widely adopted as a software modeling notation
- Modelica is very useful for hardware modeling

- Problem: Size and imprecise semantics of UML
- Possibility: Eclipse offers a minimal and well defined UML-like base platform:
  EMF (Eclipse Modeling Framework)

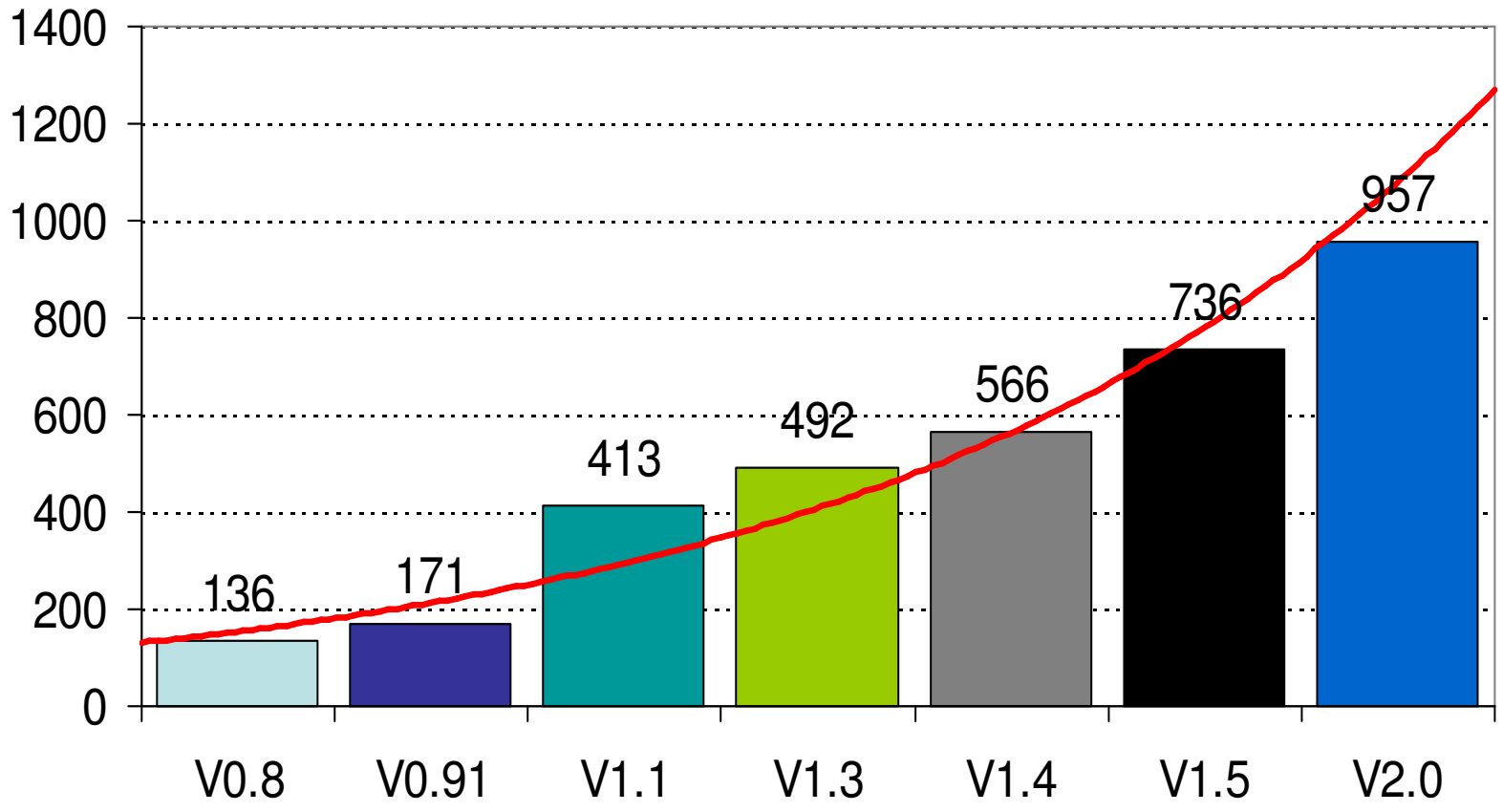
UML Suitability Problems

- Model Exchangeability
- Specification Size
- Semantics
- Sub-Languages (OCL, Action Language)
- Incompatible Children (SysML, xtUML)

UML is not Exchangeable



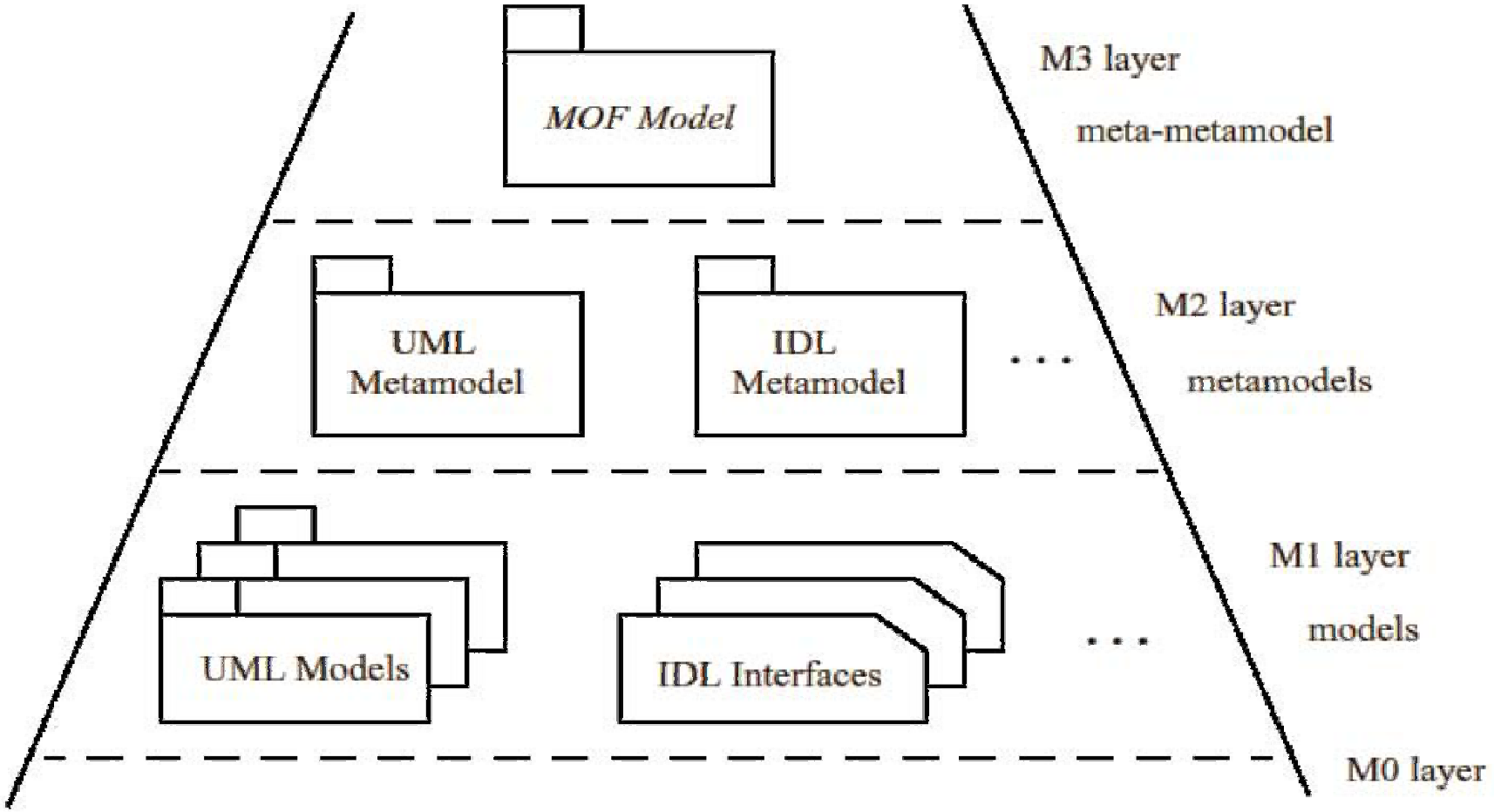
UML Specification Growth (Pages)



What is Metamodelling?

A Software Engineering approach
to cheaply build interactive editors
for engineering languages that are
described in the form of class diagrams
using diagrams to specify/code compilers.

UML (Meta-Object Facility) MOF Pyramid



[UML1.4.2]

Structure and Consistency Based on MOF

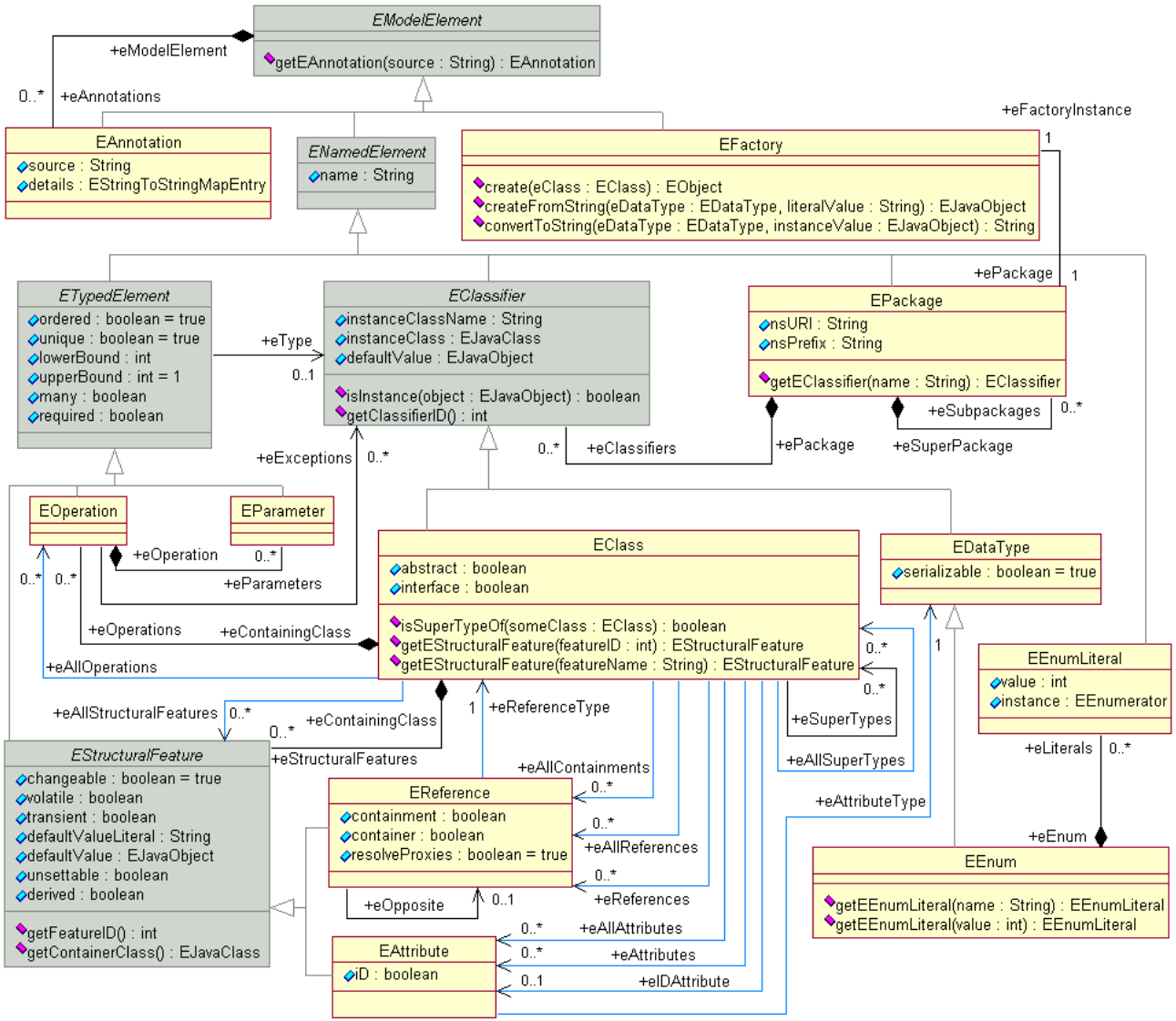
- Classes
- Attributes
- References
- Multiplicities
- Well-formedness Rules

- Eclipse Ecore
  MOF Model Instance

MOF Advantages

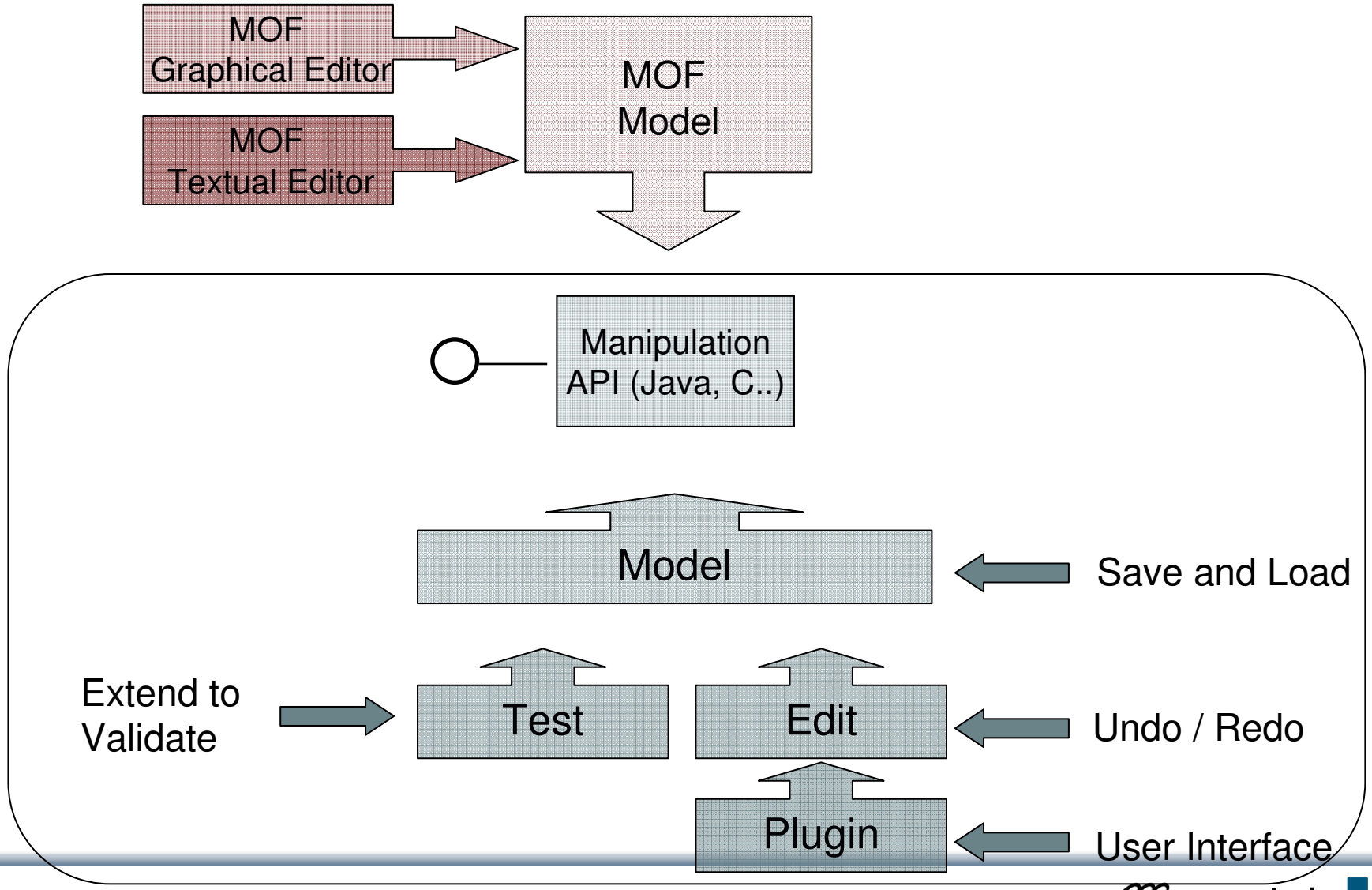
- High-level Semantics (Boxes and Lines)
  - Less (visible) technology
  - Easier to understand for client
- Generic Builder Tools (Graphical Editors)
- Large Provided Infrastructure / Framework
  - Reduced cost to build solution
  - Simplified integration between solutions
- Report Generators
  - Produce Work Items
  - Produce Documentation

Defining the MOF Language in MOF

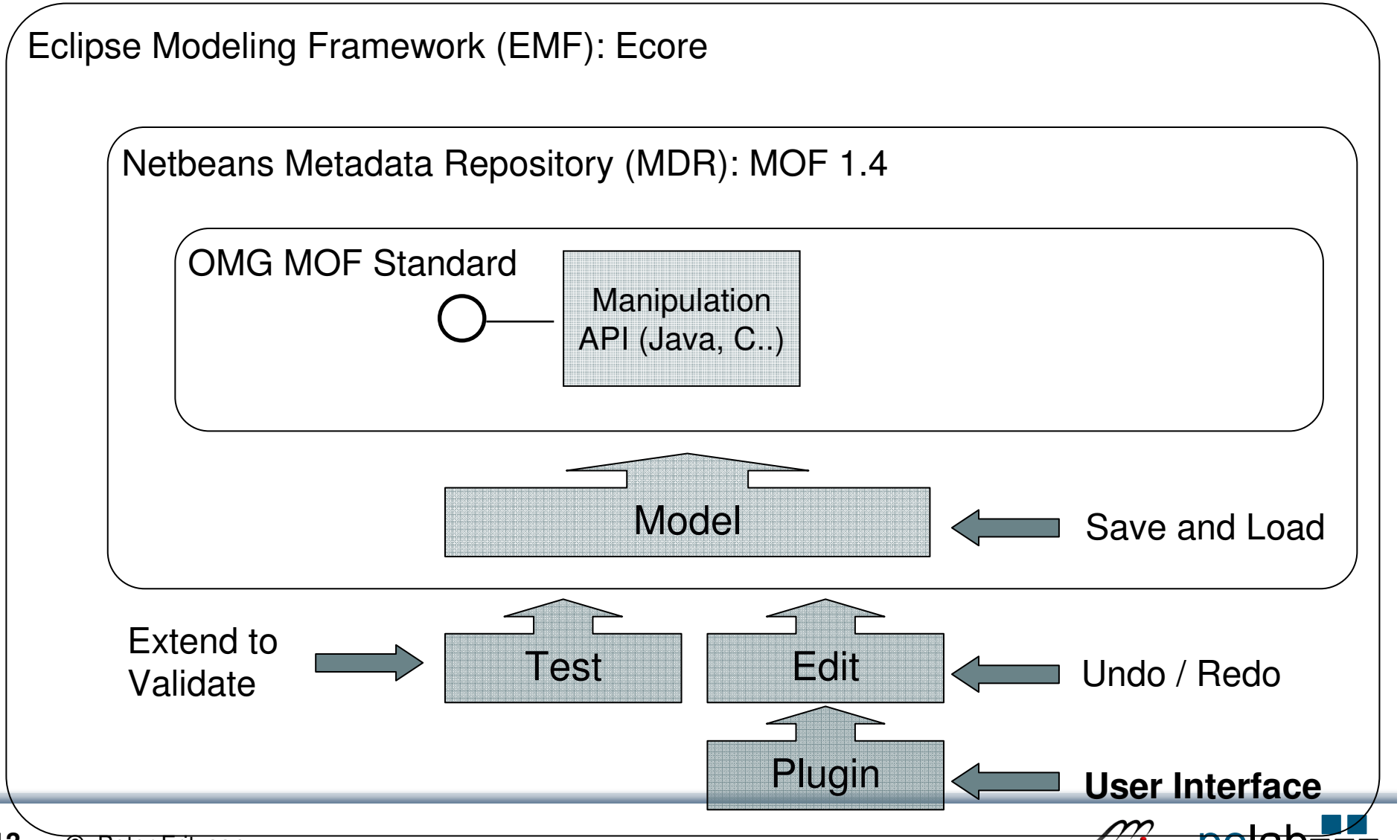


[MOF1.4]

MOF Translation Process



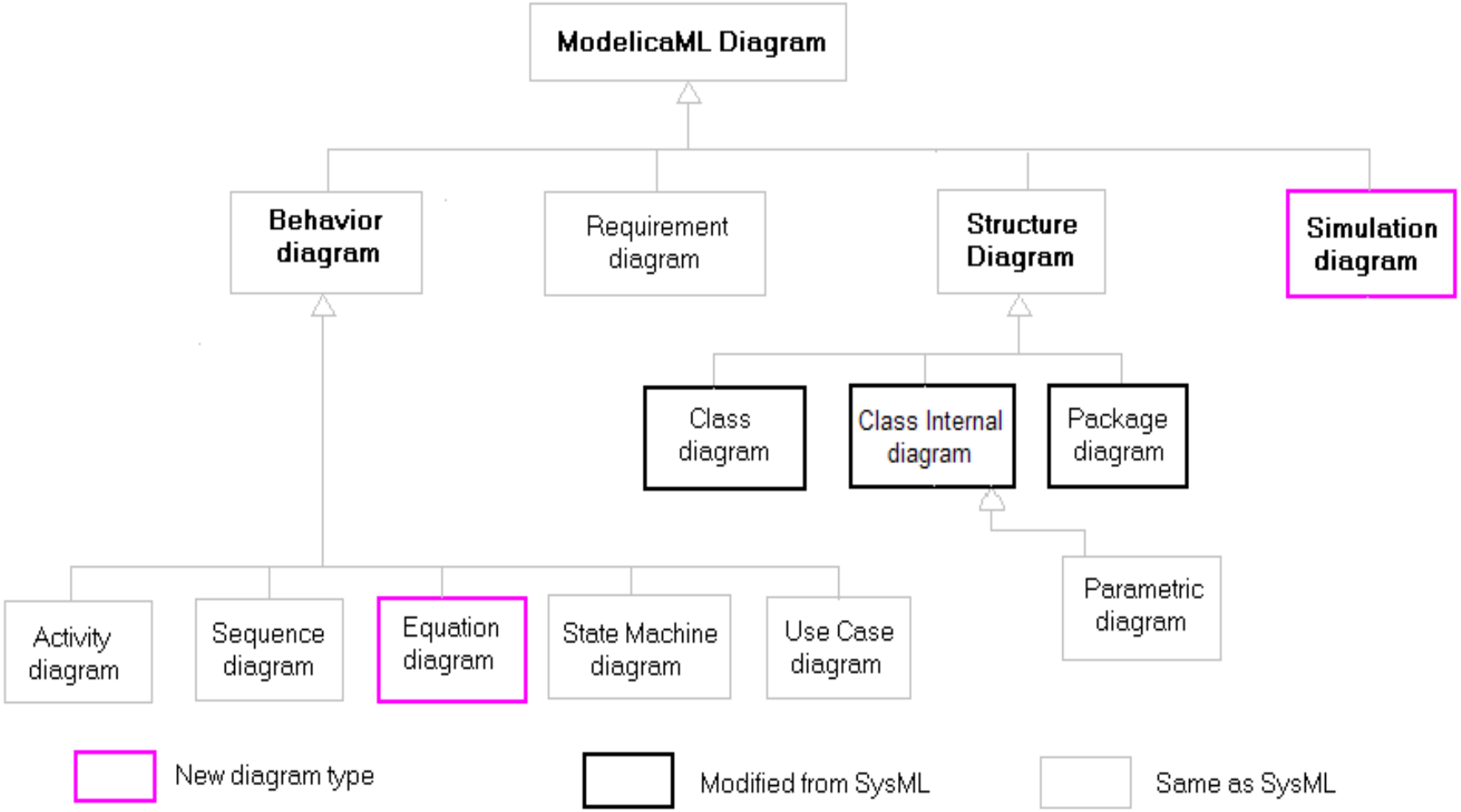
MOF Framework Offerings



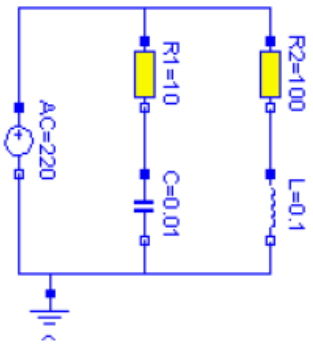
Ongoing Work: ModelicaML – UML Profile for Modelica

- Extension of SysML subset
- Features:
  - Supports Modelica constructs
  - Modelica generic class modeling
  - Modelica syntax in definitions
  - Equation-based modeling
  - Simulation modeling

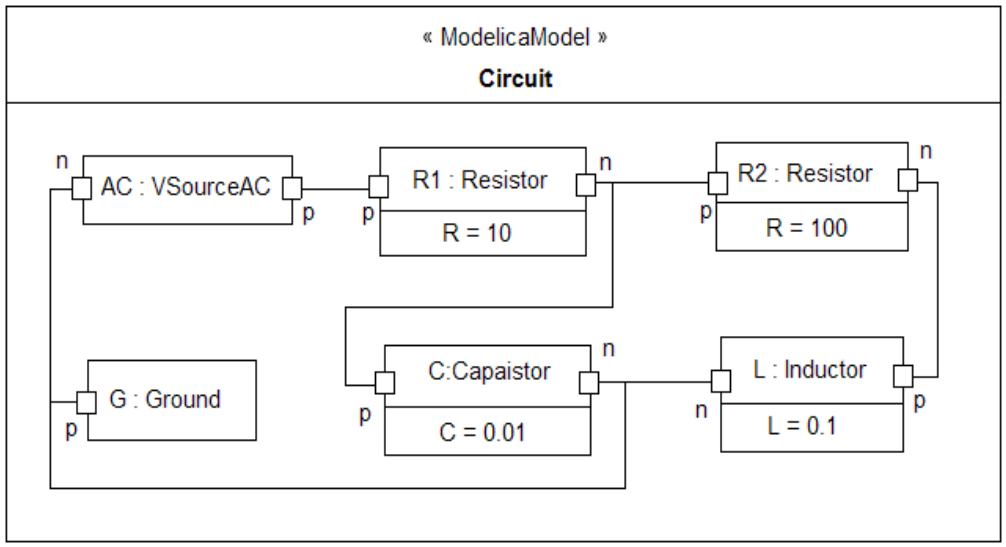
ModelicaML Diagrams – Overview



ModelicaML Class Internal Diagram

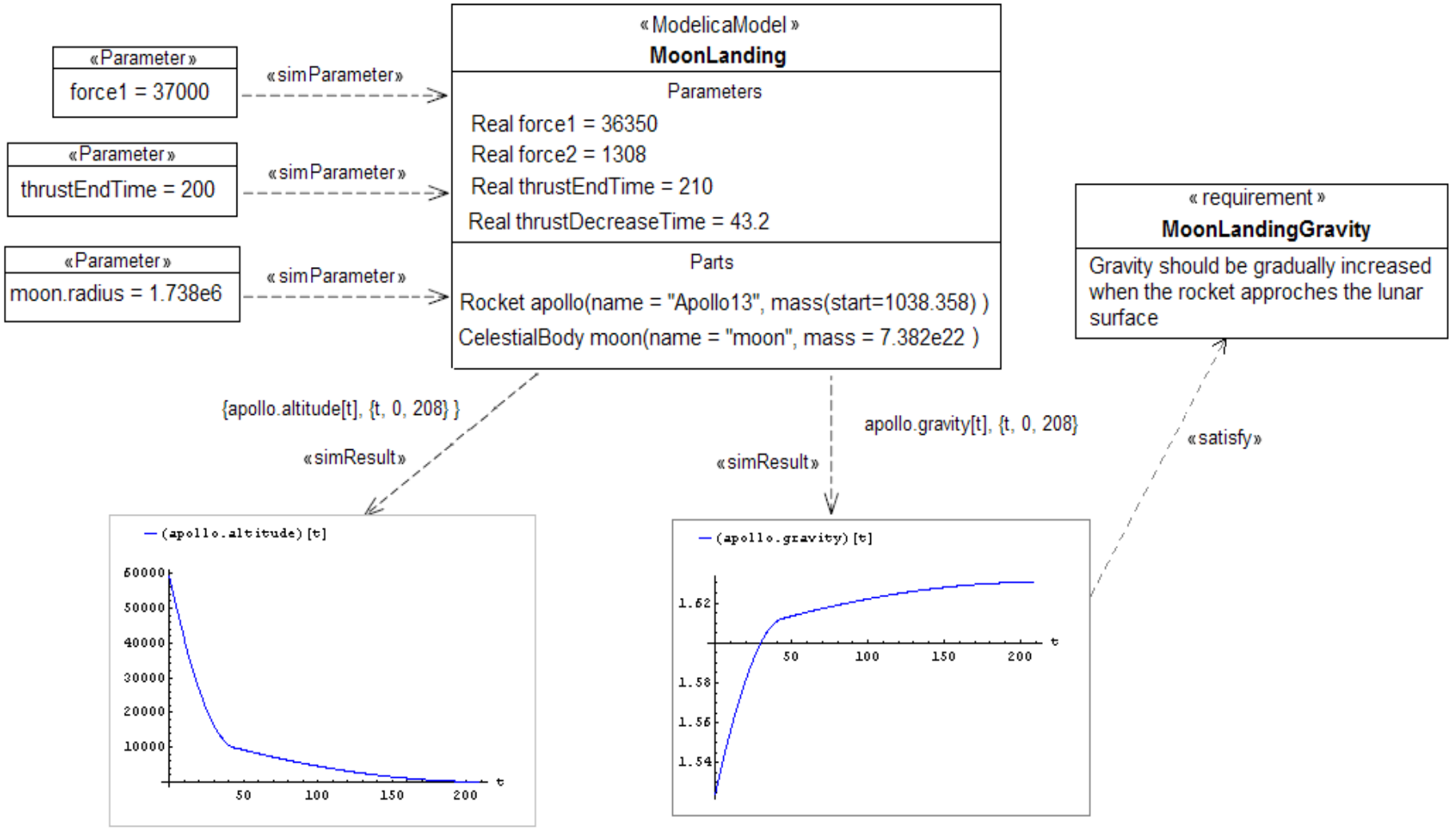


versus

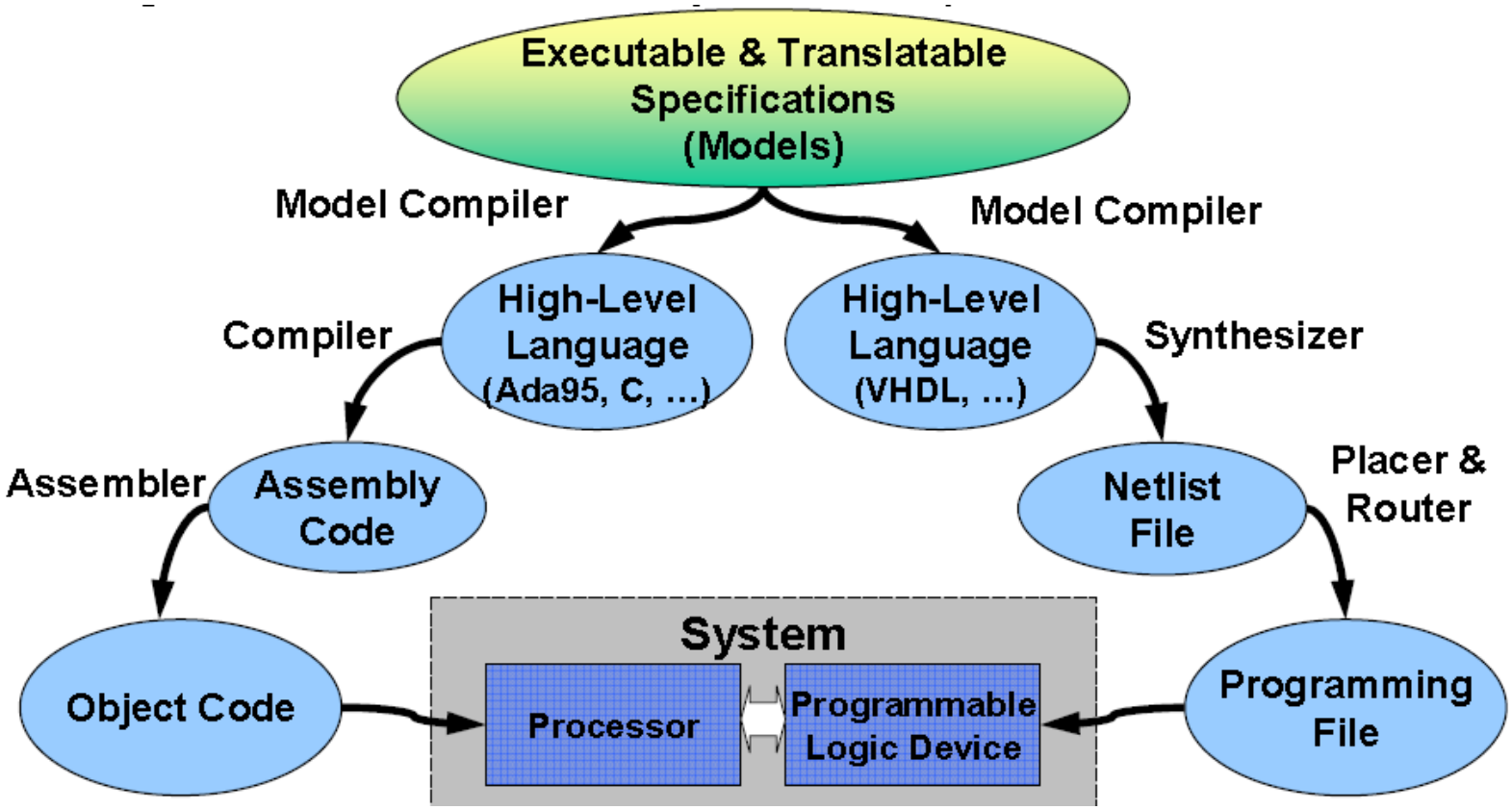


- Modelica Connection diagram
  - Better visual comprehension
  - Predefined connector locations
- Class Internal diagram
  - Nested models
  - Top-model parameters and variables
  - Flow direction
  - Other ModelicaML elements

Simulation Diagram Introduced by ModelicaML

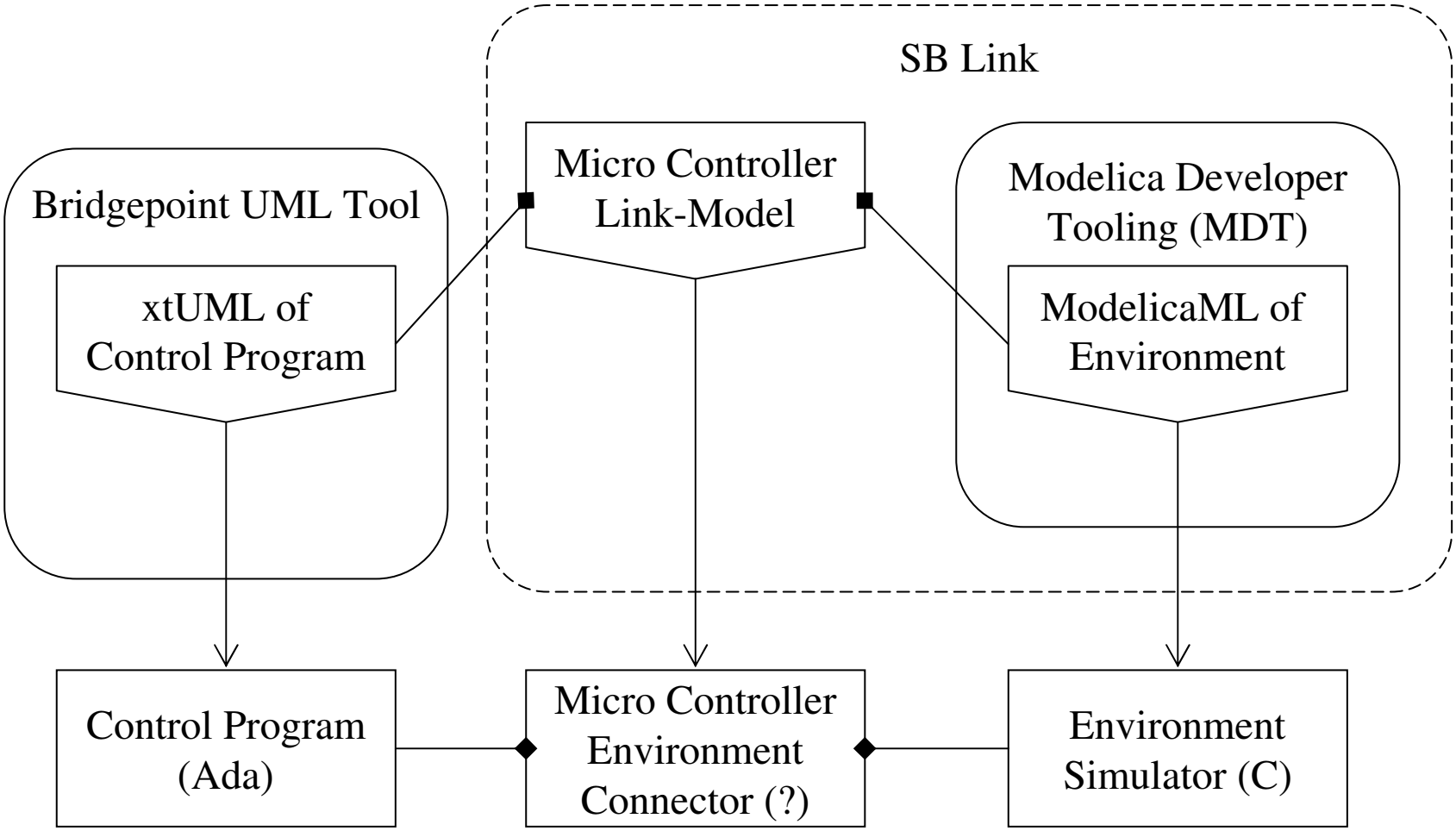


Saab Bofors Example Application



[SAAB BOFORS]

Future Tool Integration Example
(Saab-Bofors application)



Outlook for ModelicaML

- Continue using Eclipse Modeling Framework as a basis
  - Based on Ecore meta-meta-model
- Make ModelicaML smaller with more well defined semantics
  - Only include semantically well-defined diagrams
  - Remove some UML/SysML constructs with fuzzy semantics
  - Full compilation to Modelica
- Use algorithmic Modelica as (UML) action language?