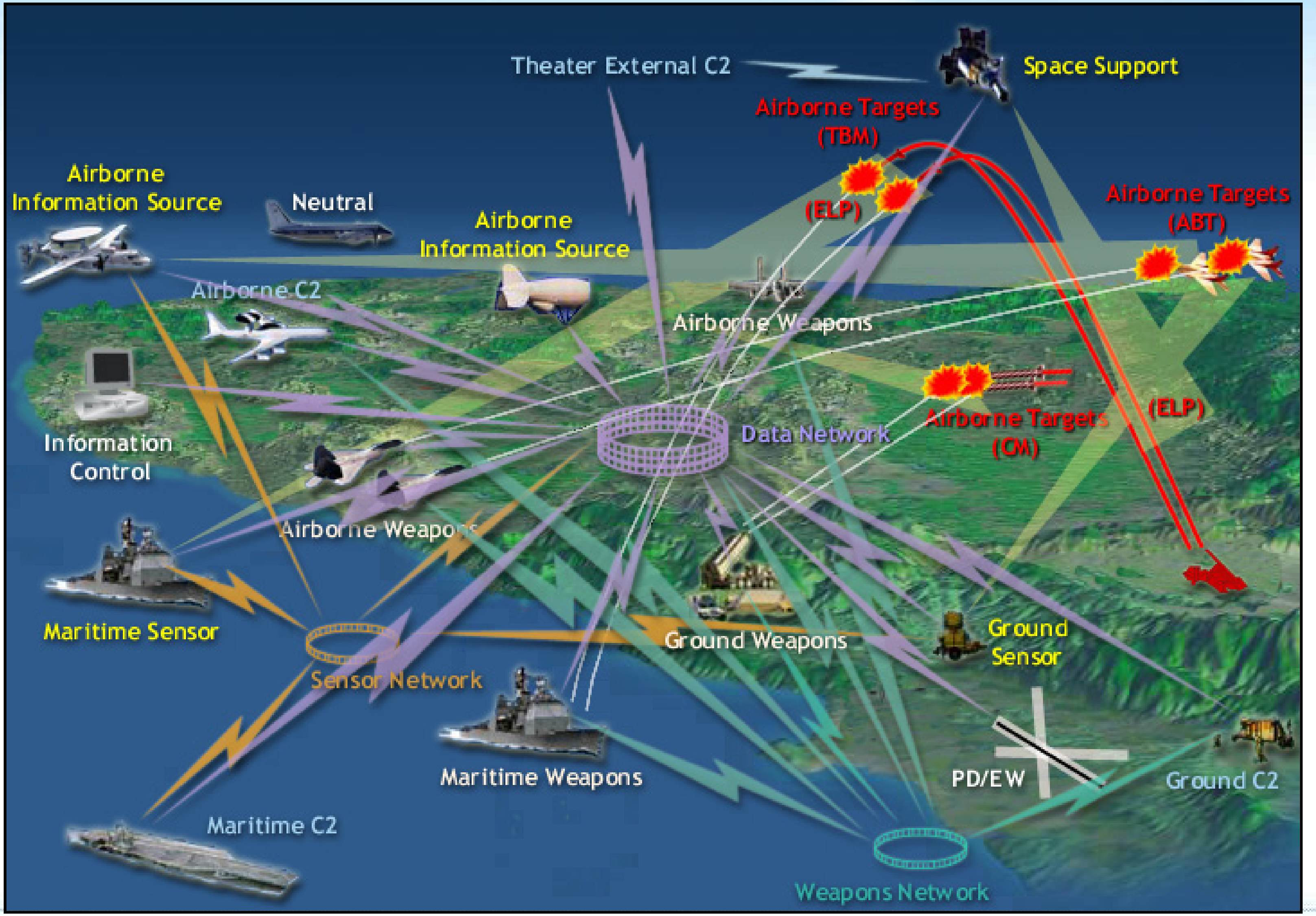
A Journey through the Secret Life of Models

(A Play in Three Acts)

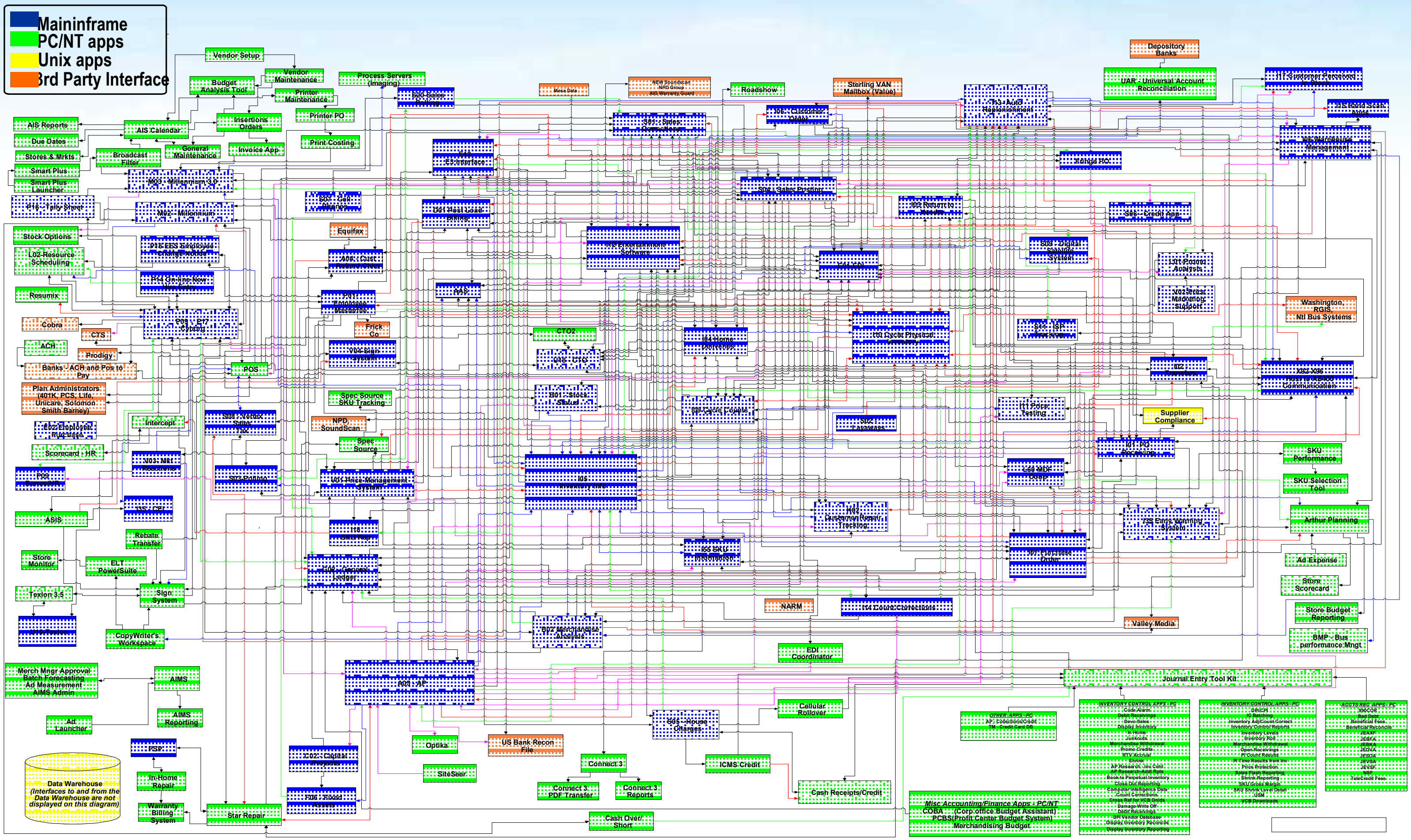
Act I – The Problems

Complexity (i)



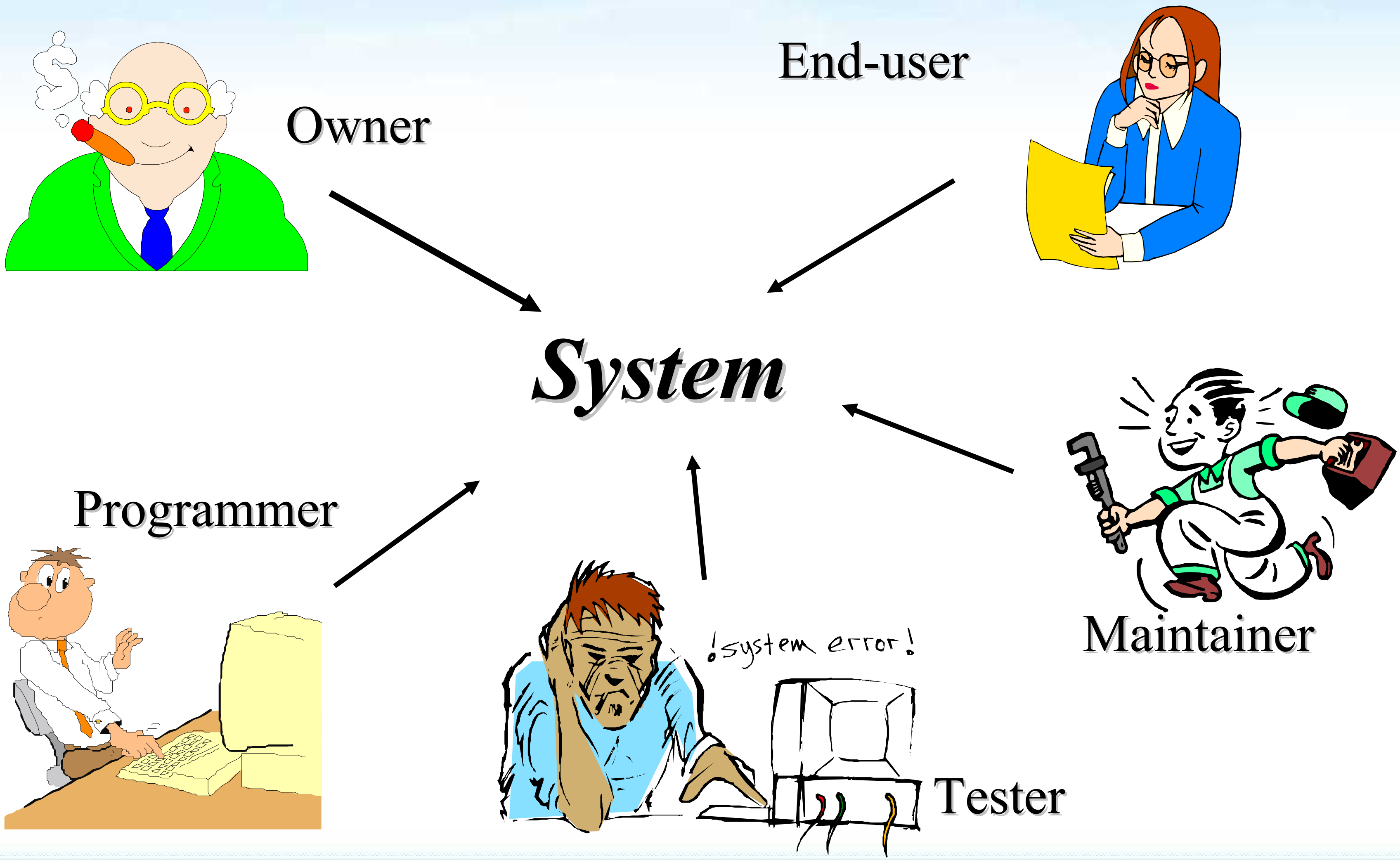
[Borrowed from Dov Dori's Tutorial on SysML Modeling at TOOLS 2008]

Complexity (ii)

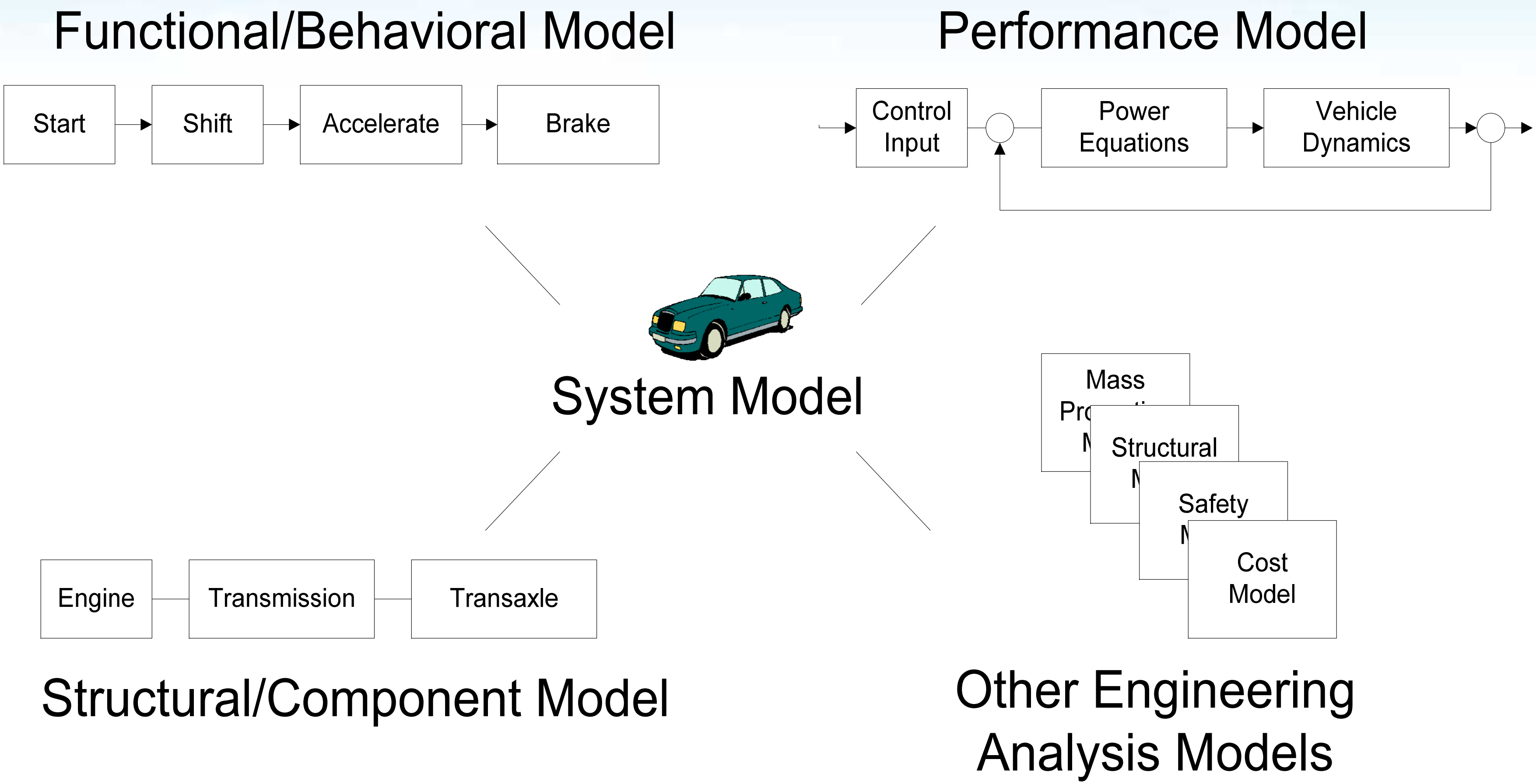


Design of a real Retail application

Different stakeholders’ viewpoints (SoC)



Multiple aspects of a system. Consistency



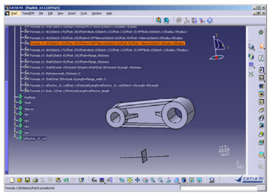
[Borrowed from Dov Dori's Tutorial on SysML Modeling at TOOLS 2008]

Lack of (integrated) analysis tools

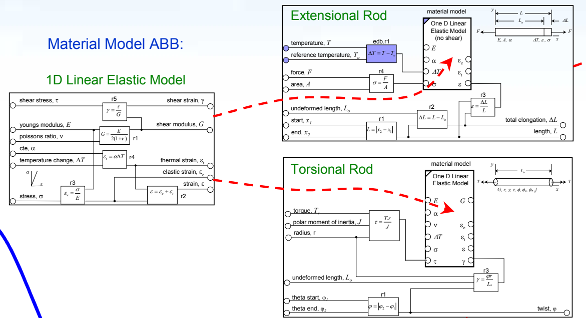
Design Tools

MCAD Tools

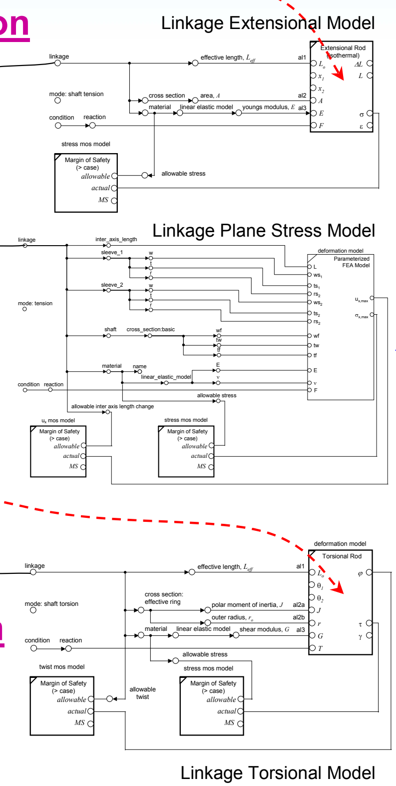
CATIA, NX, Pro/E, …*



Analysis Building Blocks *(ABBs)*



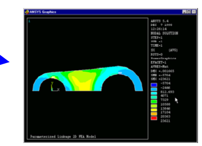
Analysis Templates of Diverse Behavior & *Fidelity* *(CBAMs)*



Analysis Solvers *(via SMMs)*

General Math

Mathematica
*Matlab**
*MathCAD**
…

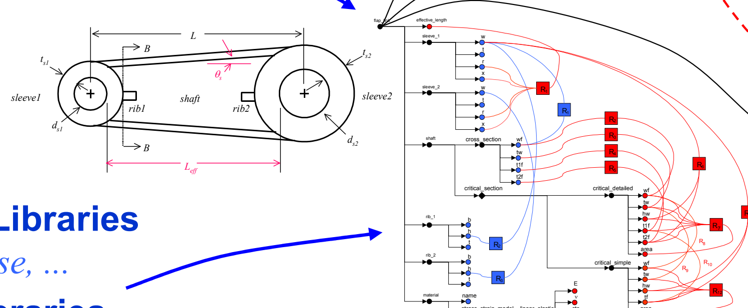


FEA

Ansys
*Abaqus**
*CATIA Elfini**
*MSC Nastran**
*MSC Patran**
*NX Nastran**
…

Legend

- Tool Associativity
- Object Re-use



Analyzable Product Model *(APM)*

Materials Libraries

In-House, …

Parts Libraries

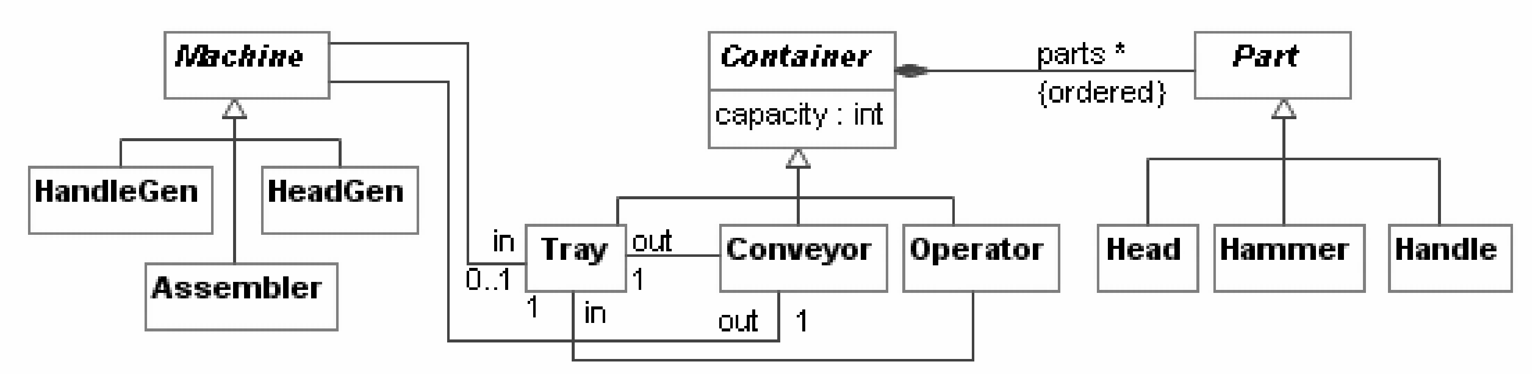
In-House, …*

* = Item not yet available in toolkit—all others have working examples 2007-04

[Borrowed from Russell Peak presentation at OMG, 2007]

Current DSLs

- Toy-ish
- Unanimated (mostly static)
- Limited analysis capabilities

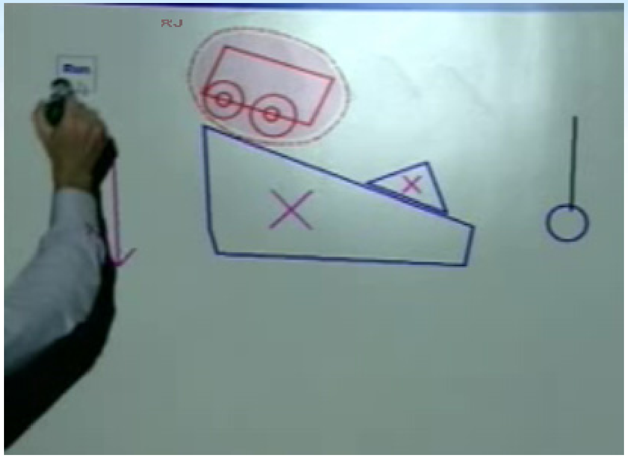


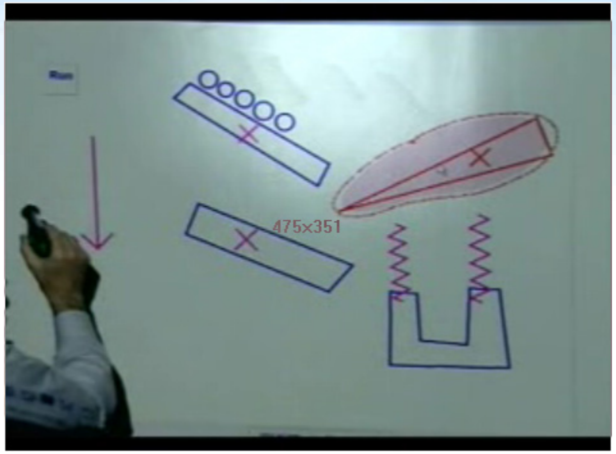
Act II – The Answers

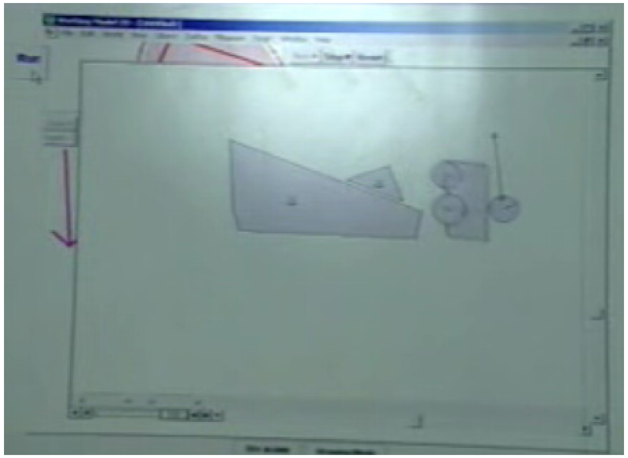
We need to be able (at least) to:

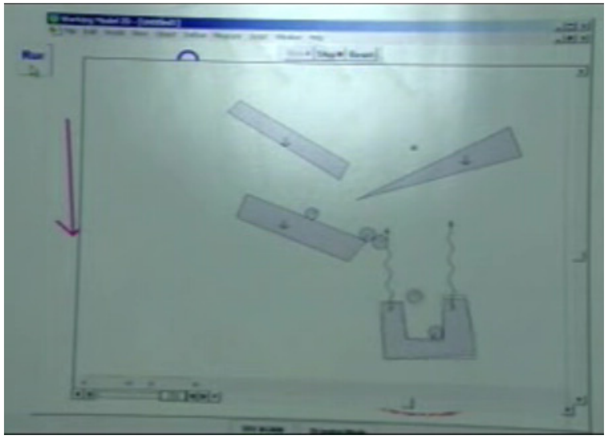
- Deal with both the **accidental** and the **essential complexity** of large-scale software systems
  - Use separate viewpoints to specify systems (each viewpoint uses its corresponding DSL)
  - Check the consistency of multi-viewpont specifications
- **Animate** models
  - Explicitly define behavioral semantics of DSLs so that models can be understood, manipulated and maintained by both users and machines
  - Define different semantics (separate concerns)
- **Analyse** models
  - Add Non-Functional Properties (Time, Probabilities,...) to DSLs
  - Connect DSLs to Analysis tools

An example of a (more useful) DSL



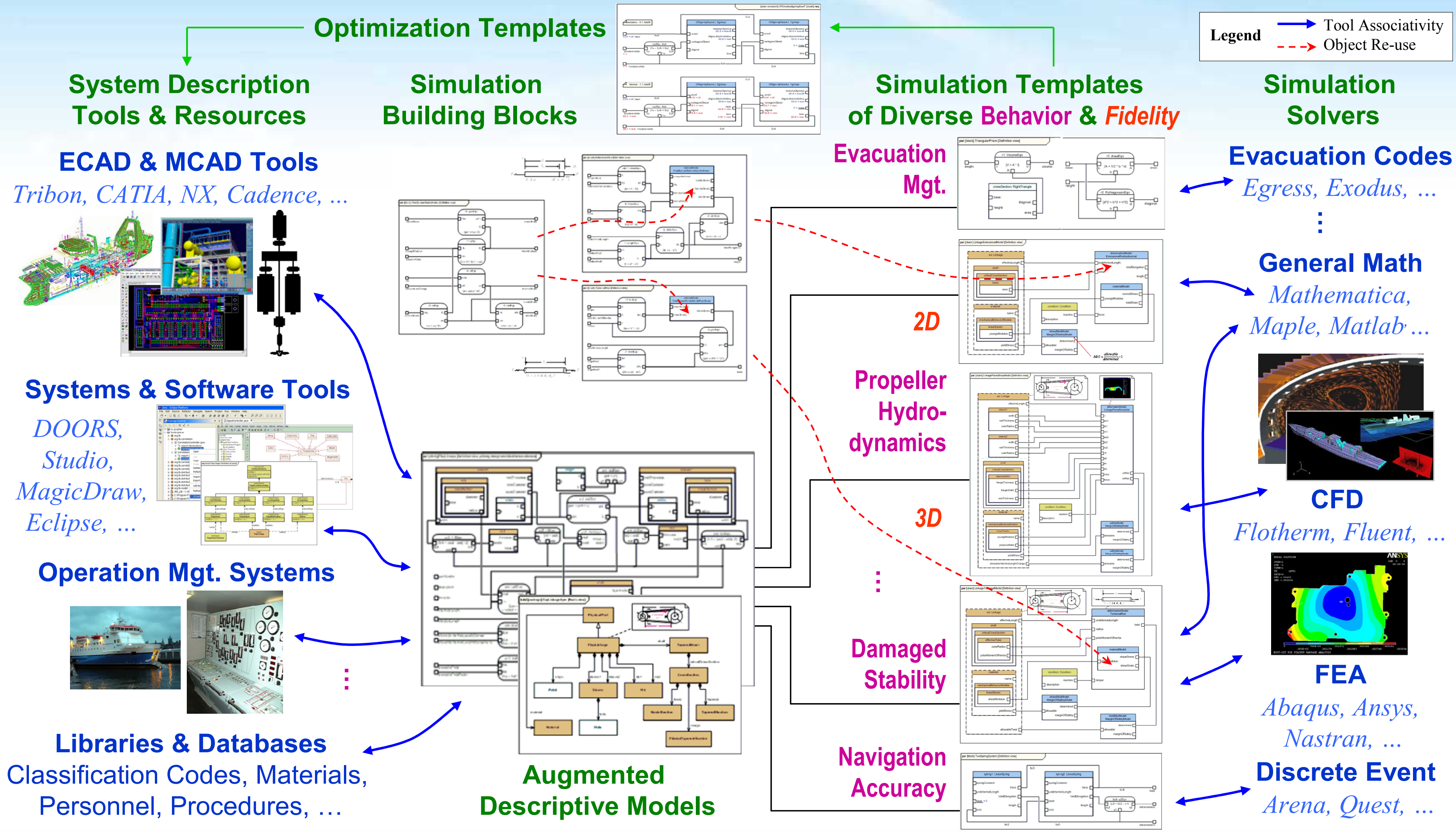






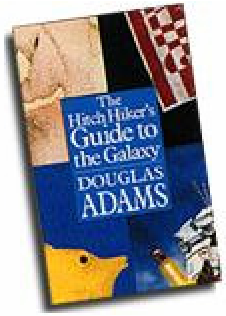
http://www.youtube.com/watch?v=NZNTgglPbUA

Use of models to connect the tools



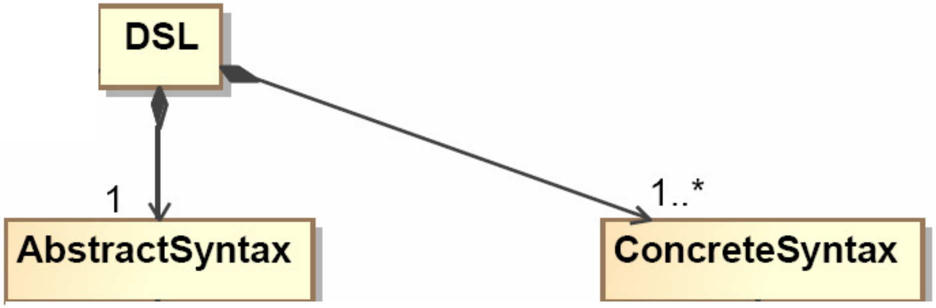
[Borrowed from Russell Peak presentation at OMG, 2007]

Act III – The Questions

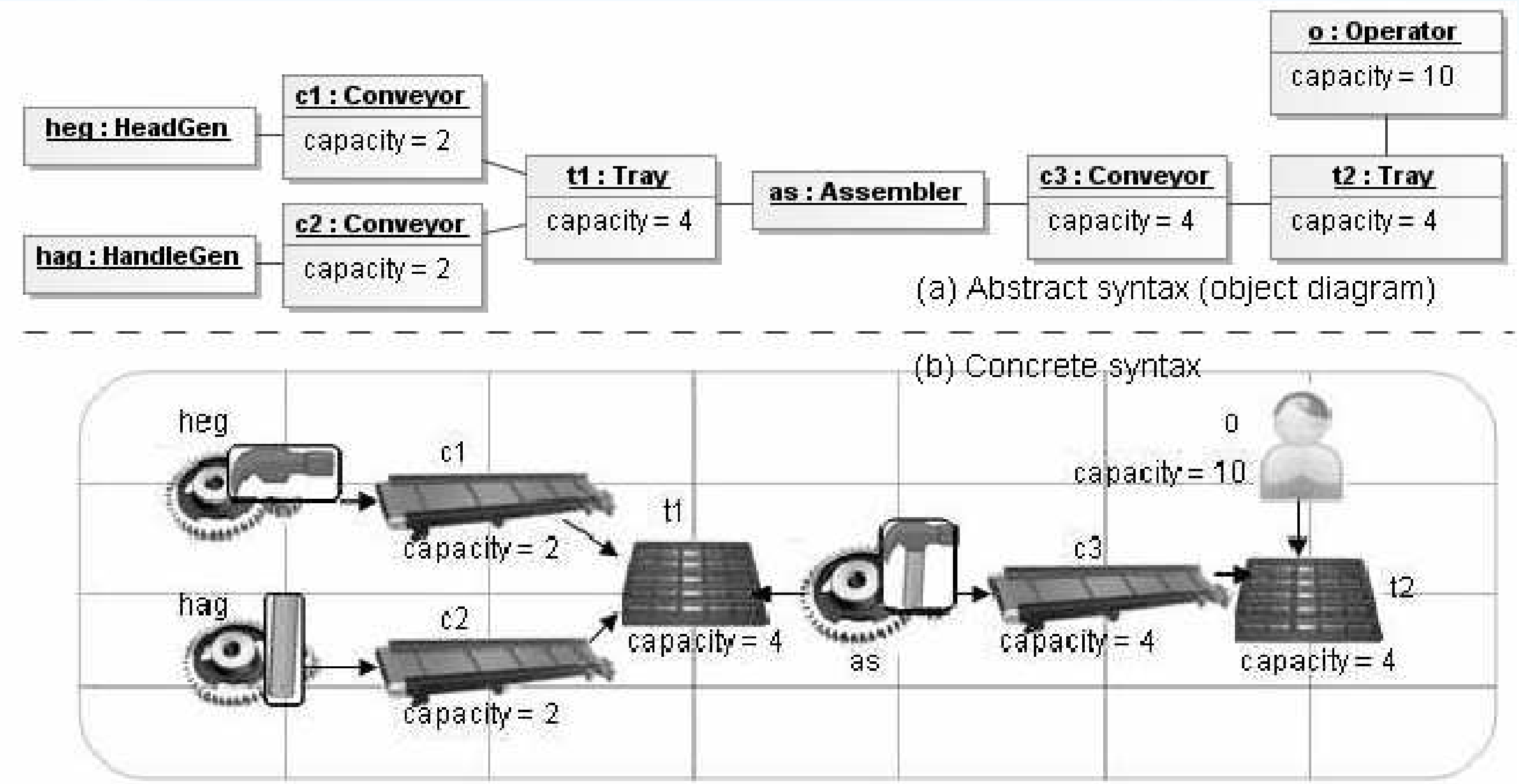


Q1. What is (in) a DSL?

Anatomy of a DSL



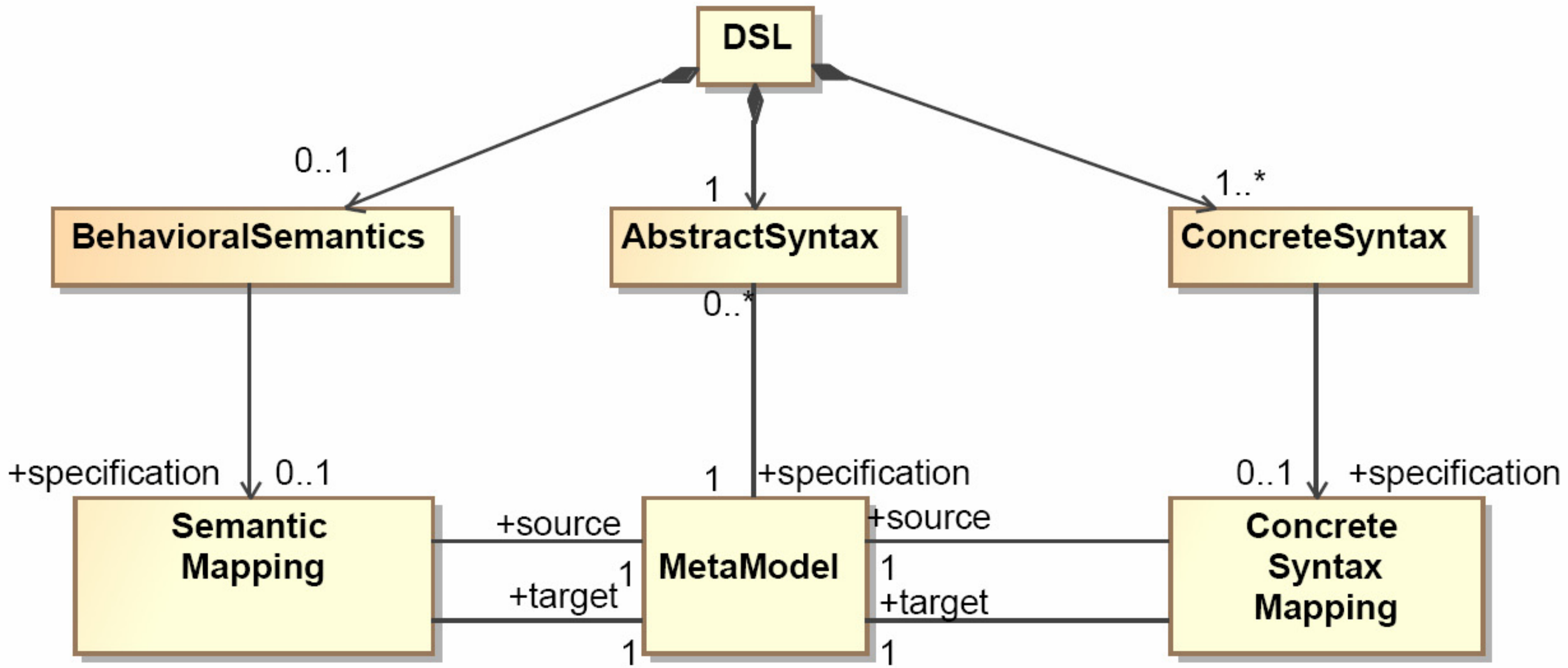
Abstract and concrete syntax



Q2. How do we add behavior?

- …to animate models (i.e., execute them)
- …to be able to conduct simulations
- …to be able to perform different kinds of (automated) analysis

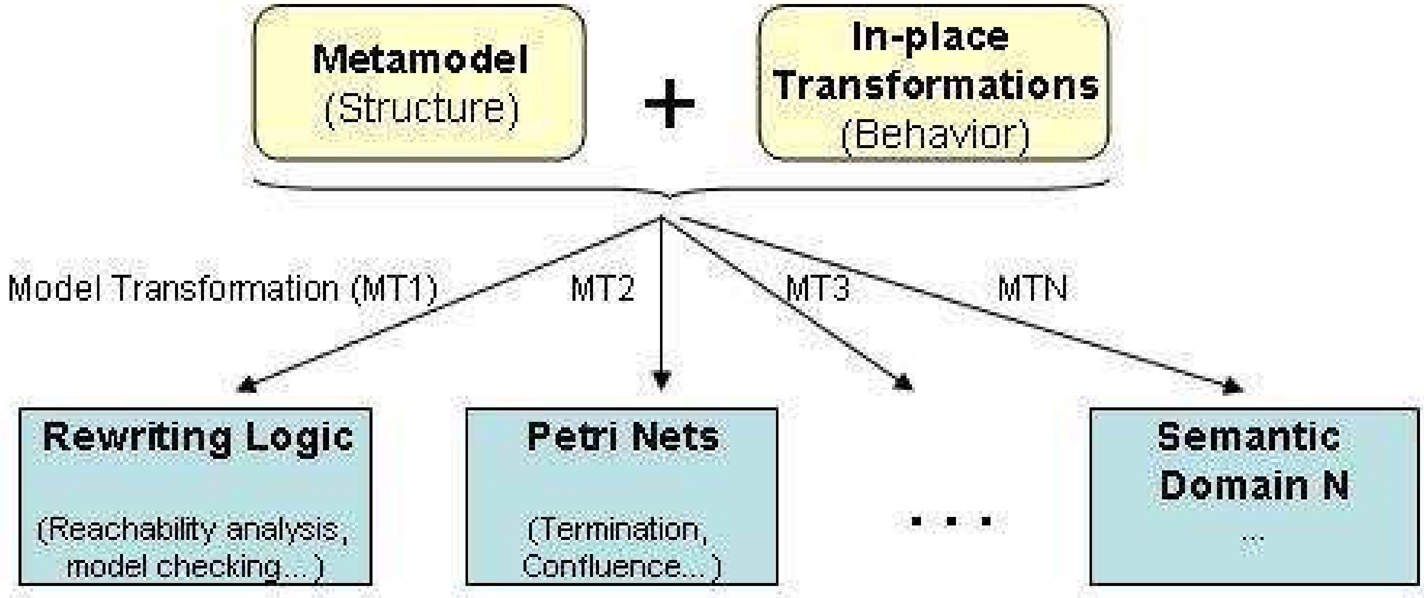
Anatomy of a DSL (II)



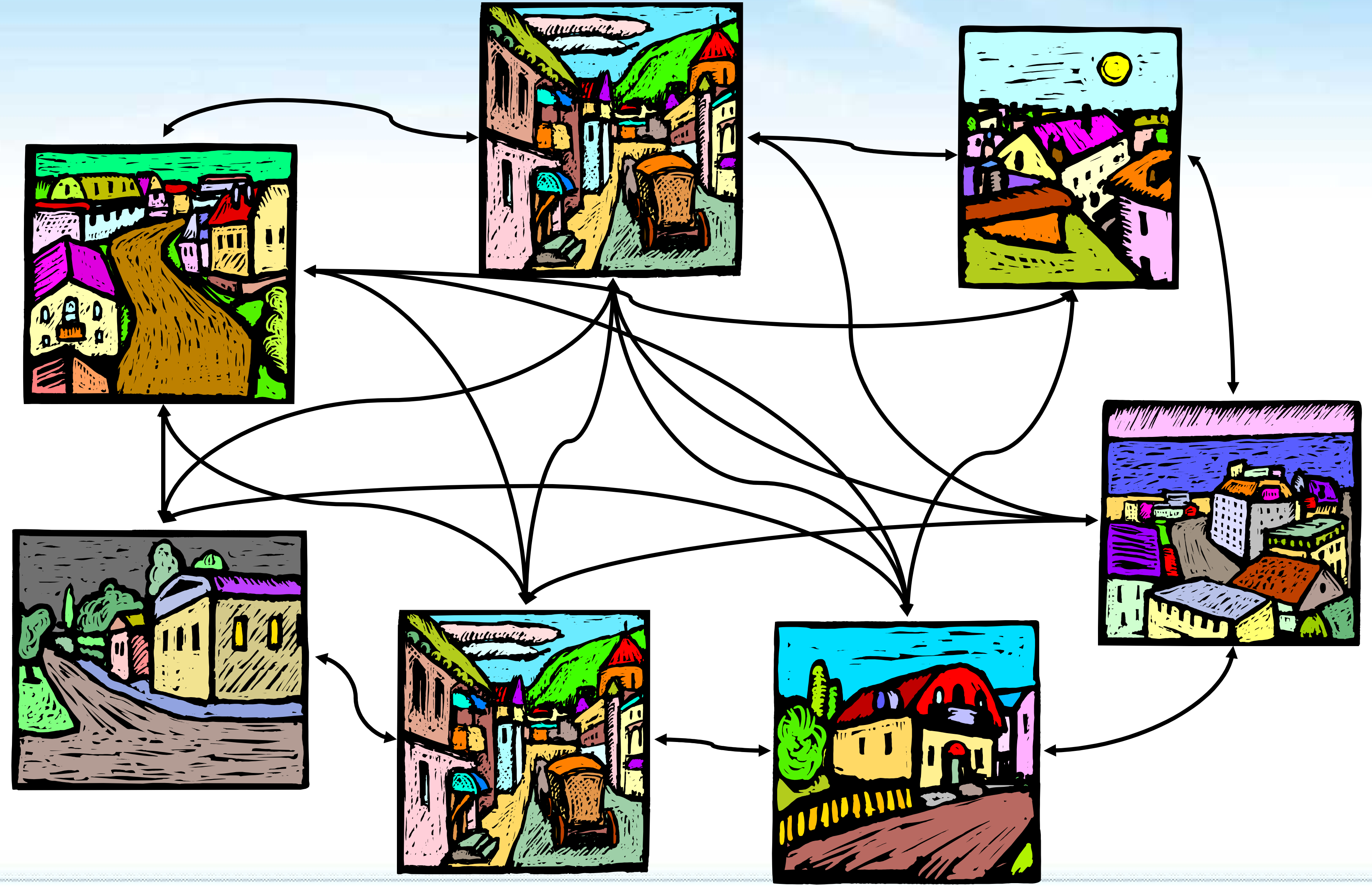
$$[\![\text{source}(e)]\!]_{\text{source}} ::= [\![\text{target}(e)]\!]_{\text{target}}$$

Semantic bridges between Semantic Domains

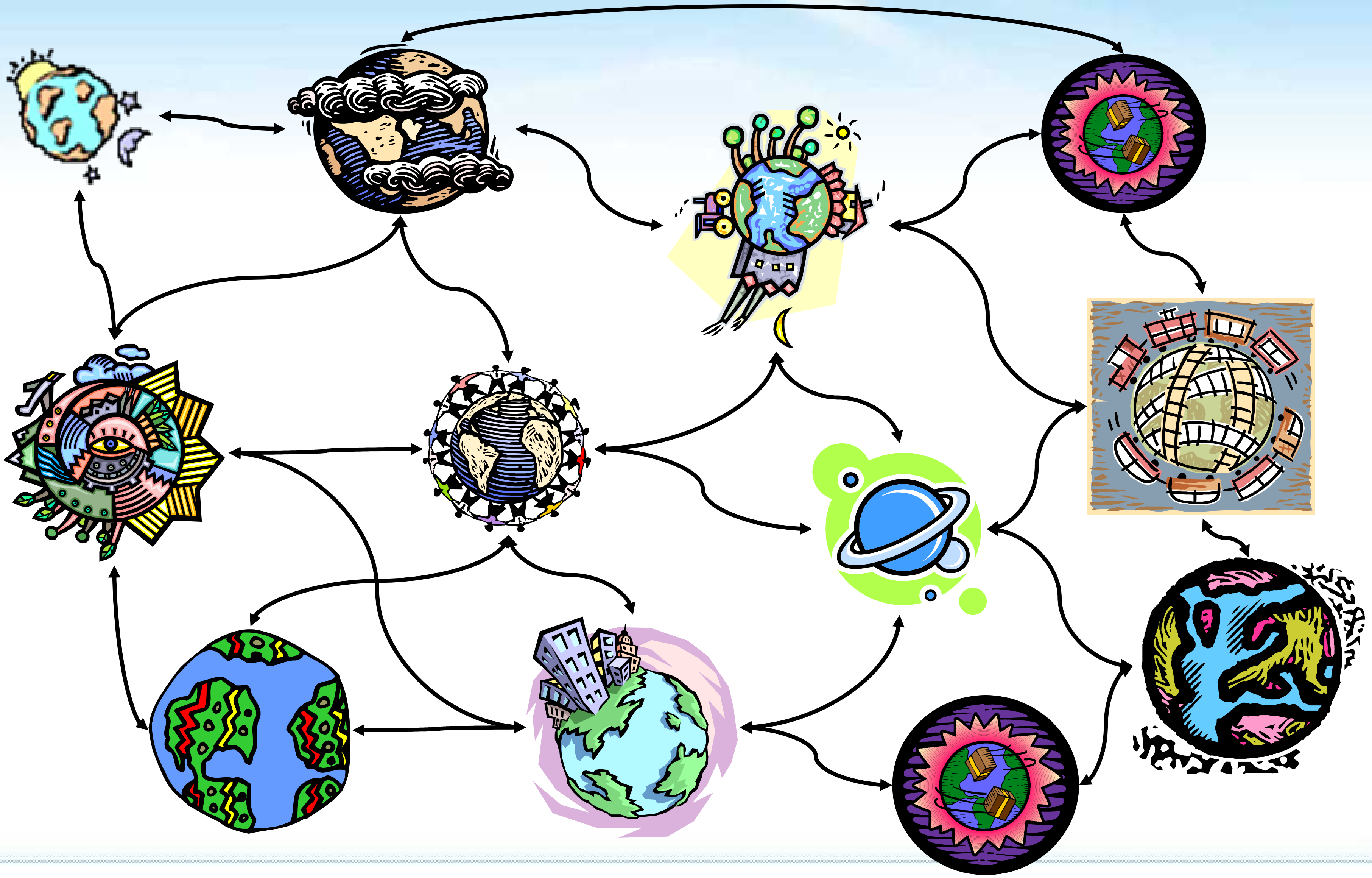
- Precise semantics
- A set of (equivalent) notations
- A set of Analysis Tools
- Underlying logic



Bridges between Semantic Domains

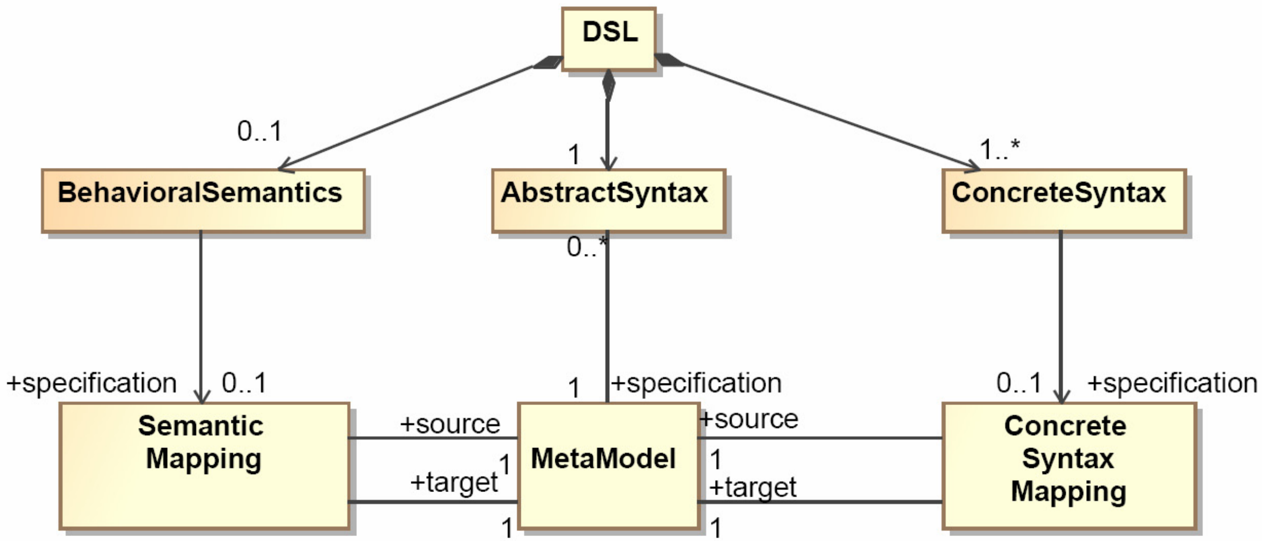


Bridges between Semantic Domains

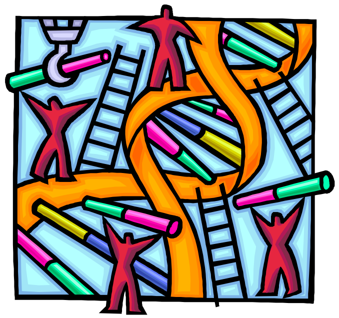


Q3. How to implement Semantic Mappings?

- As Model Transformations!!!



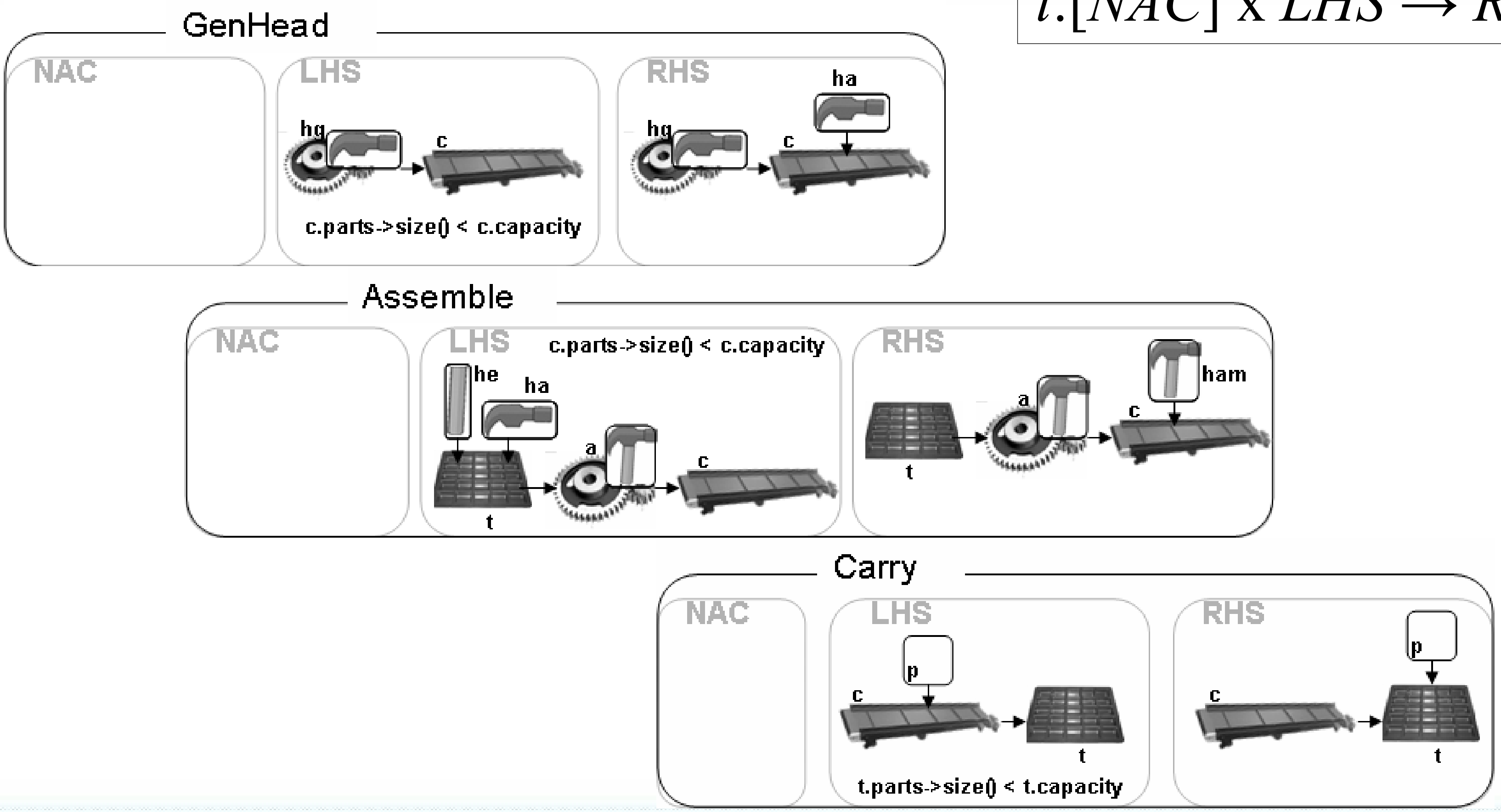
- Types
  - Domestic
  - Horizontal
  - Vertical
  - Abstracting
  - Refining
  - Pruning
  - Forgetful
  - …



Behavioral semantics

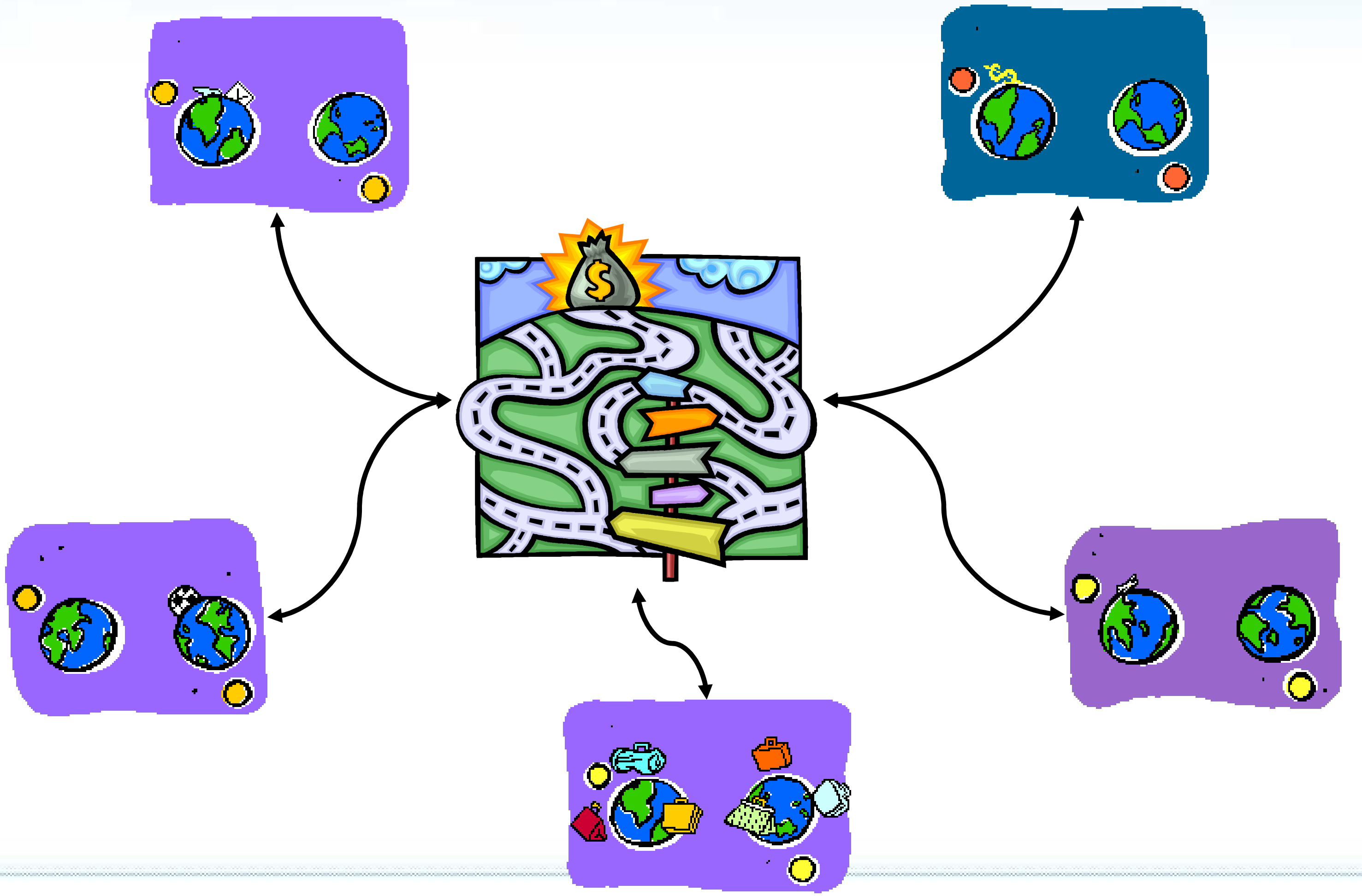
- Using in-place model transformations

$$l:[NAC] \times LHS \rightarrow RHS$$



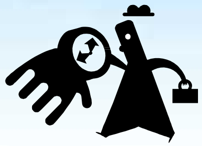
Q4. How do we analyse models?

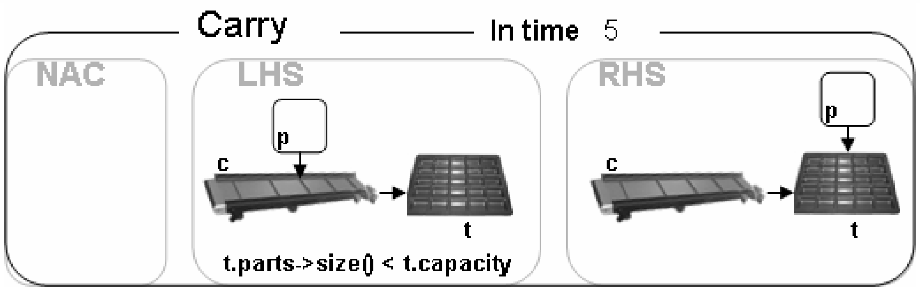
- Crossing the bridges!!!



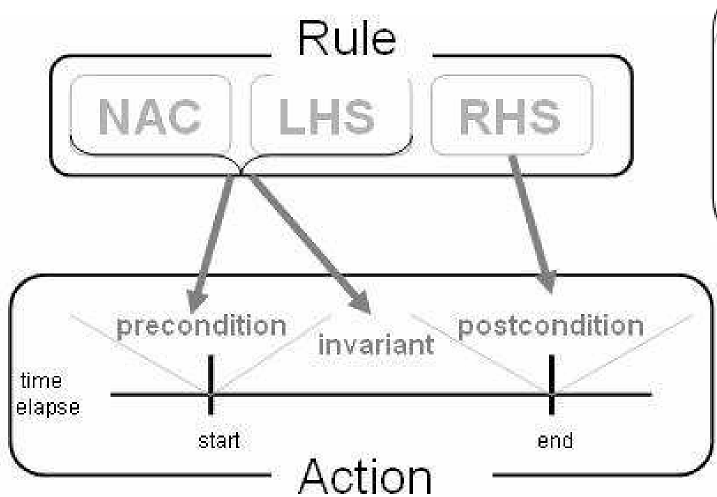
Q5. How to add time

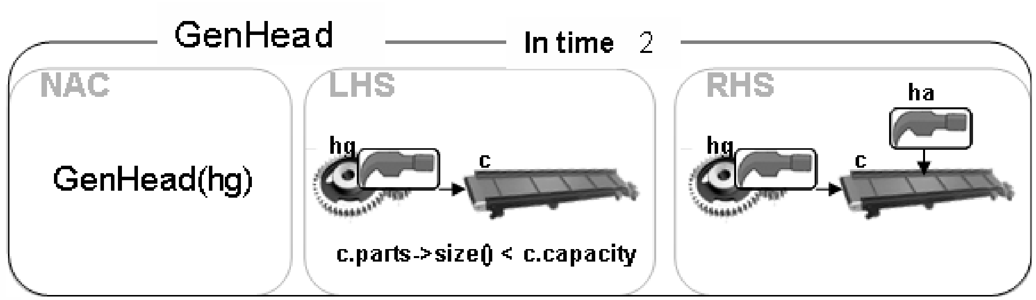
- Using in-place model transformations
- But adding the duration of the action





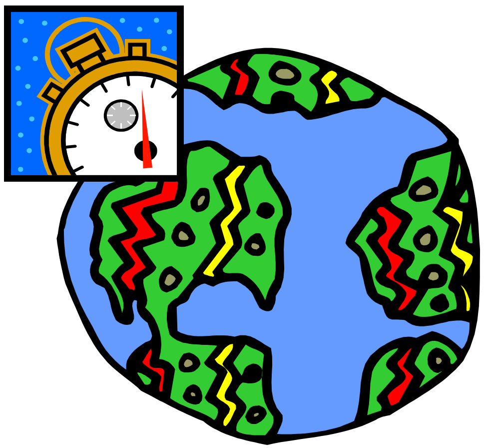
$$l{:}[NAC] \times LHS \xrightarrow{t} RHS$$





Precise Semantics of Timed Rules

- Defined by a Semantic Mapping to Real-Time Maude



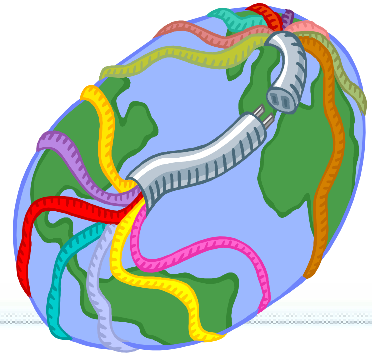
- This makes models amenable to formal analysis using the Real-Time toolkit!

More NFP required

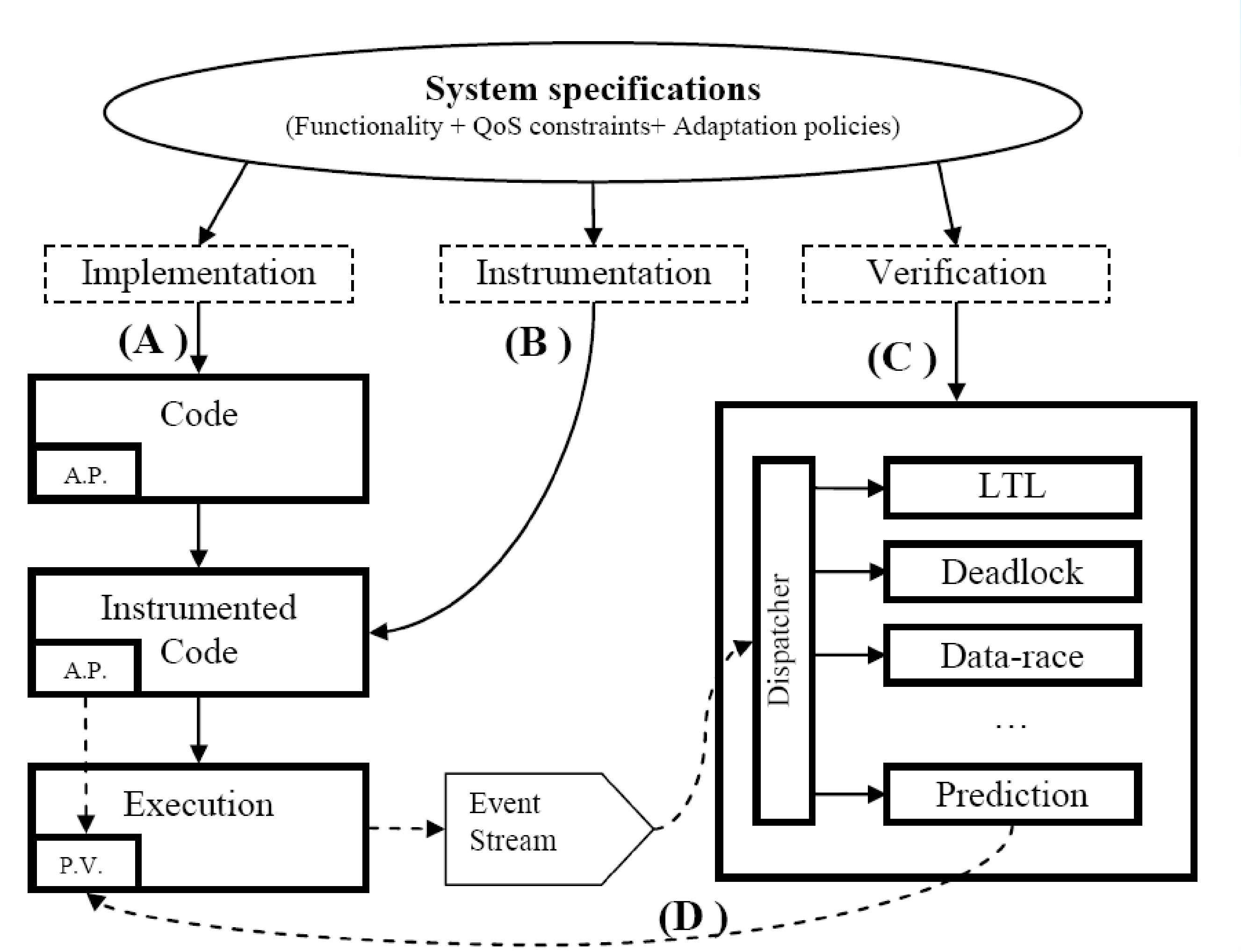
- In addition to time…

- Probabilities
- Resource consumption
- SLAs
- …

- How to add them to our behavioral specifications?
- How to connect them to existing analysis tools?



Model-driven Run-time monitoring



[MDD-MERTS Spanish project TIN2008-03107, 2009-2011]

Q6.What is a Multiviewpoint Specification

Definition 1 (Initial) *A* System Specification *consists of a set of views* $V = \{V_1, \ldots, V_n\}$. *Each view* V_i *is a model that conforms to a metamodel* $\mathcal{M}_i$ *(the viewpoint language).*

- This is the approach used by most EAFs
- No correspondences between the viewpoint elements…
  … or trivially based on name matching
- Others assume the existence of a global metamodel

A global metamodel

- Easier to manipulate from a theoretical point
- Simplifies reasoning about consistency

BUT...

- The granularity and level of abstraction of the viewpoints can be arbitrarily different
- The viewpoints may have very different formal semantics
- Should it consist of the intersection or of the union of all viewpoints elements?
  - Both approaches have serious problems with extensibility and expressiveness (not to mention complexity of the second approach – think in the UML 2.0 metamodel).

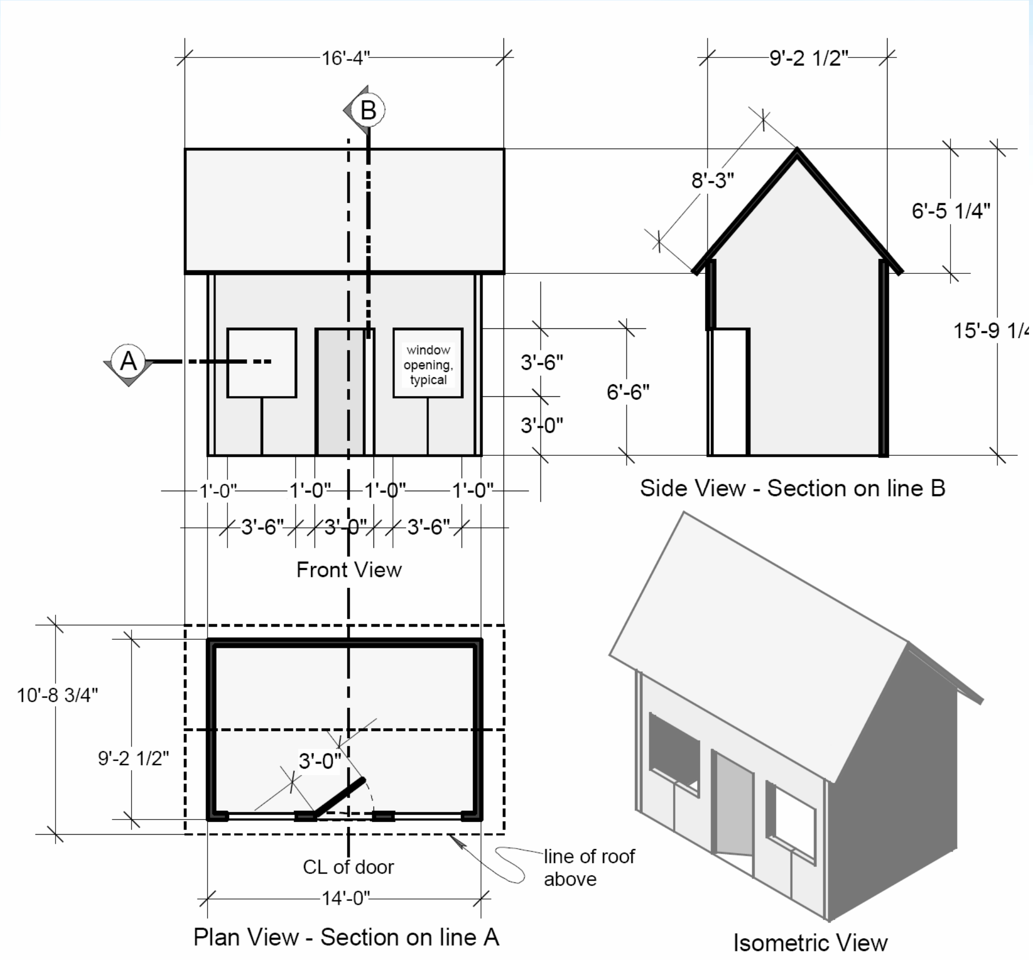
A global metamodel
(i.e., Sauron’s approach to UML)

The lord of the Metamodels

(obviously, adapted)

Three notations for the Structure modelers under the sky,
Seven for the Behavior modelers in their halls of stone,
Tree for Mortal Men doomed to die,
One for the Designer of the Whole system on his dark throne
In the Land of Mordor where the Shadows lie.
One Metamodel to rule them all, One Metamodel to find them,
One Metamodel to bring them all and in the darkness bind them
In the Land of Mordor where the Shadows lie.

Correspondences: Orthographic projections



Multiviewpoint Specification

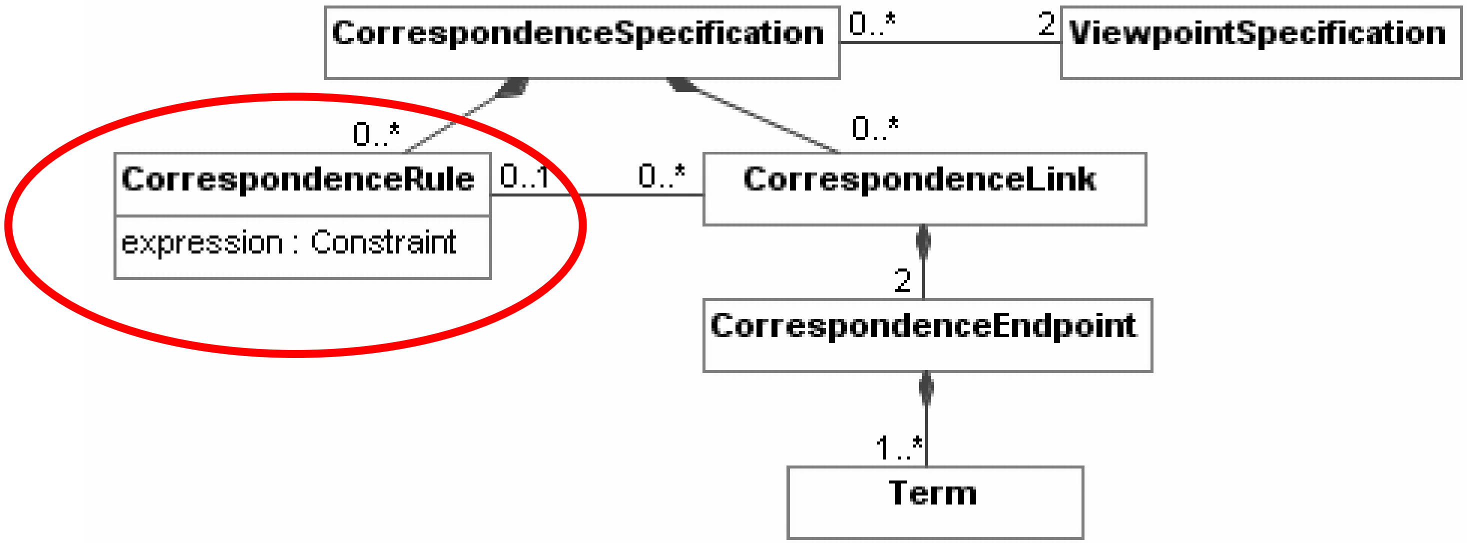
Definition 1 (Initial) *A* System Specification *consists of a set of views* $V = \{V_1, \ldots, V_n\}$*. Each view* V_i *is a model that conforms to a metamodel* $\mathcal{M}_i$ *(the viewpoint language).*

Definition 2 (With explicit correspondences) *A* System Specification *consists of a set of views* $V = \{V_1, \ldots, V_n\}$ *and a set of correspondences* $C = \{C_{(1,2)}, C_{(1,3)}, \ldots, C_{(n-1,n)}\}$ *between the views. Each view* V_i *is a model that conforms to a metamodel* $\mathcal{M}_i$ *(the viewpoint language). Correspondences are also models, and each* $C_{(i,j)}$ *conforms to a correspondence metamodel* $\mathcal{C}$*.* [1]

Expressing correspondences

- As **Model Transformations**
  - Possible if correspondences can be expressed as functions
  - Pairwise consistency can be formally studied
    - One form of consistency involves a set of correspondence rules to steer a transformation from one language to another. Thus given a specification S_1 in viewpoint language L_1 and specification S_2 in viewpoint language L_2, a transformation T can be applied to S_1 resulting in a new specification $T(S_1)$ in viewpoint language L_2 which can be compared directly to S_2 to check, for example, for behavioral compatibility between allegedly equivalent objects or configurations of objects [RM-ODP, Part 3]
- As **Weaving Models**
  - Possible if correspondences are just mappings

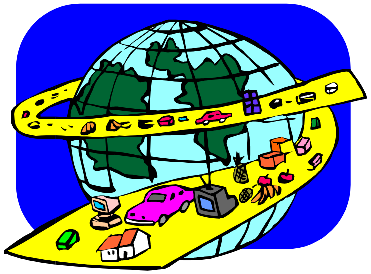
ODP Correspondence metamodel



Correspondences are not enough

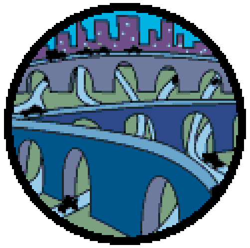
Definition 3 (With well-formed correspondences)
A System Specification *consists of a set of views* $V = \{V_1, \ldots, V_n\}$, *a set of correspondences* $C = \{C_{(1,2)}, C_{(1,3)}, \ldots, C_{(n-1,n)}\}$ *between the views, and a set of rules* $R = \{r_1, \ldots, r_k\}$ *that describe the constraints that the correspondences of* C *should fulfil in order for a specification to be well-formed. Each view* V_i *is a model that conforms to a metamodel* $\mathcal{M}_i$ *(the viewpoint language). Correspondences are also models, and* $C_{(i,j)}$ *conforms to a correspondence metamodel* $\mathcal{C}$. *Rules are expressed as constraints on the correspondence elements, using any constraint language (e.g., OCL).*

Epilogue



A Hitchhiker's Guide to Metamodels

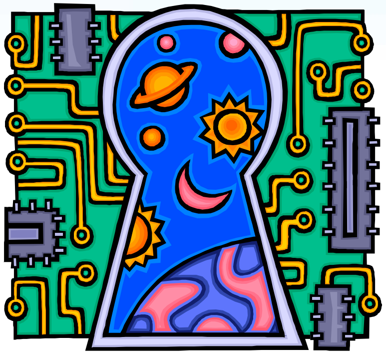
- Use **multiview** specifications of systems
  - Composed by a set of **Views**
  - Each view focuses on one concern
  - Each view is expressed using a Viewpoint Language (DSL)
- Views are related using correspondences for **consistency** checking
  - **Correspondences** can be defined either as model transformations or as model weavings
  - Well-formed rules should be defined for the set of Correspondences, too



- Viewpoint **DSLs**
  - Defined by an abstract syntax, a concrete syntax, and a set of semantic specifications
  - **Bridges** provide mappings to different semantic domains where models can be analyzed (using the logics and tools available at the target semantic domains)

Some more challenges

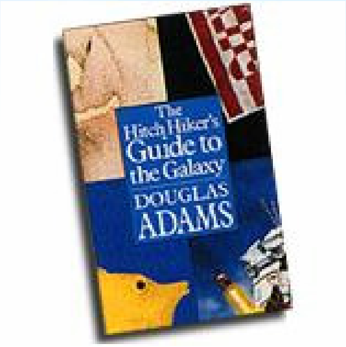
- Addition of more Non-Functional Properties for enhanced analysis capabilities
- Specification and development of more Semantic Bridges
  - Specially to semantic domains with powerful analysis tool support



- Modularity and composition mechanisms
  - Rule-based specifications become unmanageable very soon
- Global consistency checking of specifications
  - Pairwise viewpoint consistency is not enough...

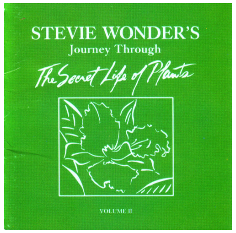
Acknowledgements

Atenea
Modeling Software Systems



GOBIERNO DE ESPAÑA

MINISTERIO DE CIENCIA E INNOVACIÓN



- And, especially, to many colleagues…