

Model Driven Design Space Exploration of Lumped-Parameter Physical Systems

Sagar Sen

McGill University
Montreal, Canada

et

TRISKELL, IRISA
Beaulieu Campus
Rennes, France

Edison's Invention of the Light Bulb



Salient Aspects of the Invention Process:

- Repetitive testing of filaments one after the other
- Knowledge directed develop and test cycle

How could Model Driven Design Space Exploration have helped Edison?

Answer: By formally representing the Invention Process



How can Model Driven Engineering (MDE) help?

1. Software can represent the **state of components** in the target system (bulb for instance) in a **model** with visual/textual syntax.
2. Software can represent the **interconnection of the components** in a model.
3. Software can represent the **engineering principles** (heuristics) that an inventor applies to improve a non-optimal model. **Model transformations** encode such heuristics.
4. Software can represent the **meaning/semantics** of individual components in the system.
5. Software can represent **constraints** on the possible state and structure of the model.
6. Software models can exist in millions and be run in parallel.

Subsuming Term: Model Driven Design Space Exploration

Design Space Exploration : Its Everywhere!

Domains: *Economics* (Eg: Administrative Behavior, Herbert Simon), *Physics* (Eg: Energy minimization), *Chemistry* (Eg: Combinatorial Chemistry), *Biology* (Eg: Protein Folding), **Engineering** (Eg: Bond Graphs, Robotics, Software test case generation) and **Scientific discovery** in general

Techniques: Genetic Algorithms, Genetic Programming, Co-evolution, Reinforcement Learning, Grammar-based Genetic Programming,...

Miscellaneous Keywords: Artificial Life, Computer Music, Parallel Design Space Search, Constraint Programming, Linear/Non-linear/Integer Programming

Terminology: Design, Automatic generation/synthesis, Planning, Decision-making, Exploration, Search,...

We do it at every step we make!

There is no unifying framework that allows you completely capture domain knowledge and clearly specify the meaning of exploration in a world of discourse !

MDE + Design Space Exploration

=

Model Driven Design Space Exploration

1. Specification of a world of discourse: **Meta-model**
2. System in this world: **Model**
3. Meaning/semantics of this model: **Model transformation**
4. Creating new models: **Heuristics** (automatically from meta-model or domain expert) as **model transformations**
5. Testing Models: **Model transformation for test case generation**
6. Constraints: **OCL constraints**
7. Visualization: A complex **graph** of objects
8. Design space search: **Any traditional method** that uses the heuristics
9. Expression of system across domains: Meta-modelling, **multi-formalism modelling** and model transformation
10. Other stuff: Contracts, aspects...

**Do we need anything else to represent any real-world system
in the software world ?**

Ideas in Engineering

You get them:

- 2) Anytime
- 3) Anywhere
- 4) Mostly Visual Models in your current framework of scientific thought

ATOM³
A Tool for Multiformalism Meta-Modelling

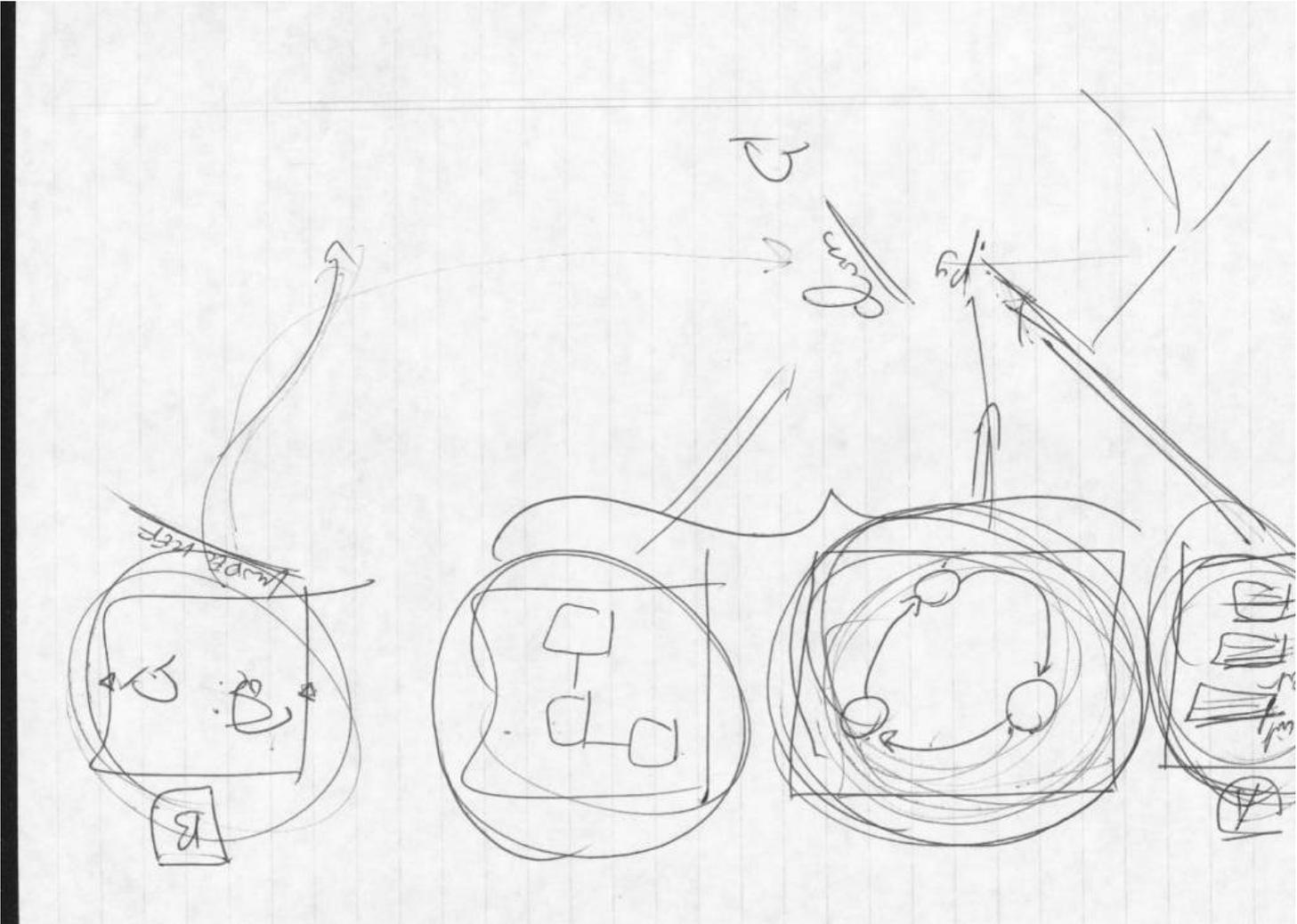
Graph Kernel

```
classDiagram
    class Node {
        #_id: string
        #_label: string
        +rename()
        +relabel()
    }
    class Graph {
        +add()
        +connect()
        +disconnect()
        +remove()
        +move()
    }
    Node <|-- Graph
```

PyGK
Morphisms
Grammars

25 10:18 AM

Scribble the idea on paper



Less often in MSDL!



Model Representation

- Graphical : Attributed graphs with python constraints, actions, specifications, pre-conditions, and post-conditions.
- Textual : Eg. Python code, Modelica code

Edge

Icon

74 Editing ACVoltageS...

function	sin
Value	110.0
freqHz	50.0
BGType	Se
Name	AC Voltage0
Type	ACSourceVoltage_E

OK Cancel

Attributes

Constraints

The screenshot displays the ATOM3 software interface. The main window shows a graphical model with a yellow circle icon representing an AC voltage source. A red arrow points from the 'Edge' label to the connection line, and another red arrow points from the 'Icon' label to the yellow circle. A dialog box titled 'Editing ACVoltageS...' is open, showing the parameters for the voltage source: function (sin), Value (110.0), freqHz (50.0), BGType (Se), Name (AC Voltage0), and Type (ACSourceVoltage_E). A red arrow points from the 'Attributes' label to the 'Type' field. Another dialog box titled 'Editing ATOM3Constraint' is open, showing Python code for a constraint. A red arrow points from the 'Constraints' label to this dialog. A third dialog box titled 'Editing Class 3' is open, showing the class definition for 'Energy_Source' with various attributes and constraints. A red arrow points from the 'Constraints' label to this dialog. The bottom of the screenshot shows the Windows taskbar with the start button and several open applications.

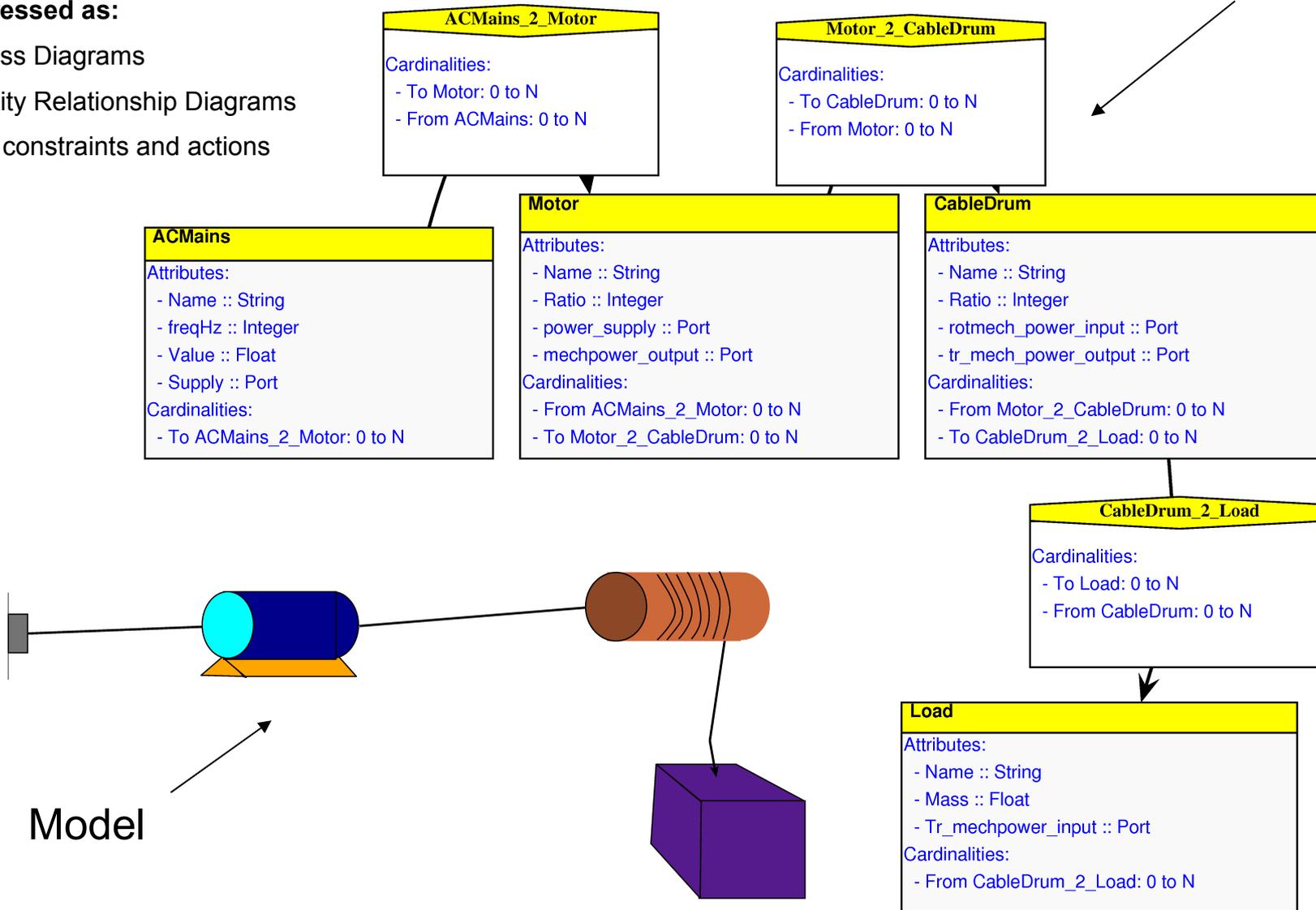
Meta-Models : To limit the modeller

Meta-model

Expressed as:

- Class Diagrams
- Entity Relationship Diagrams

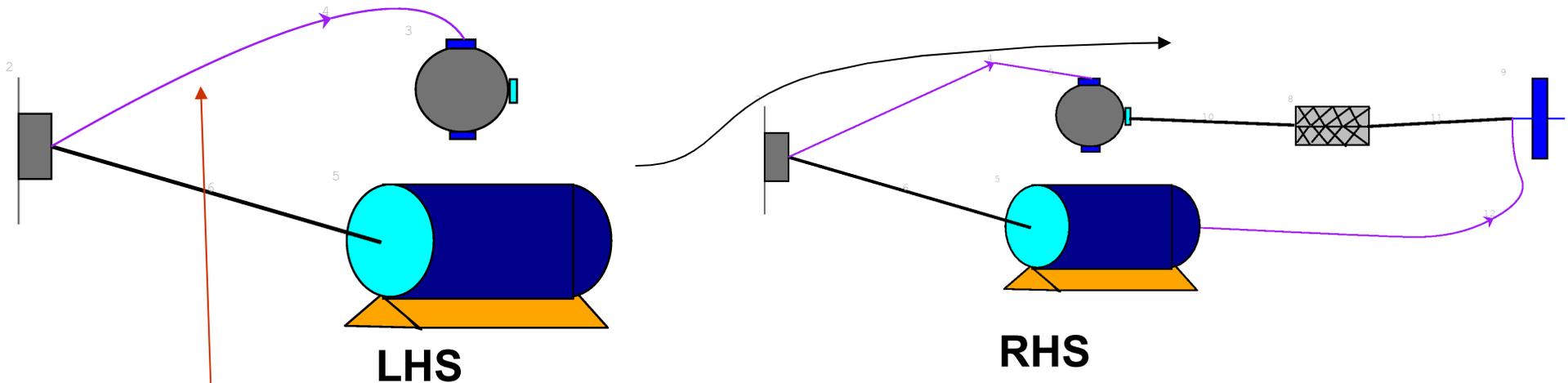
With constraints and actions



Model Transformation using Graph Grammars

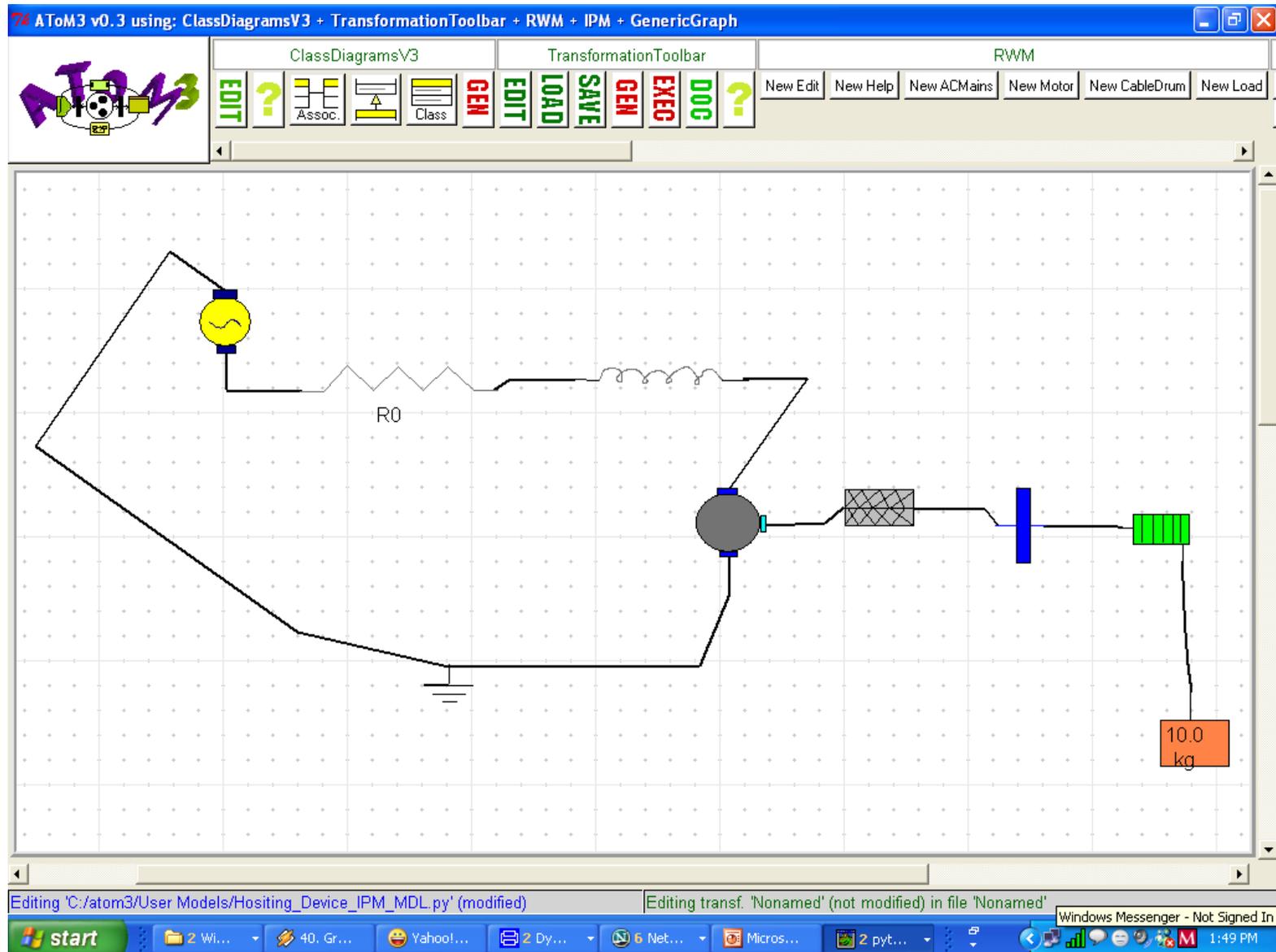
- Graph Grammars with textual constraints, actions, and specifications

Copying/specifying attributes



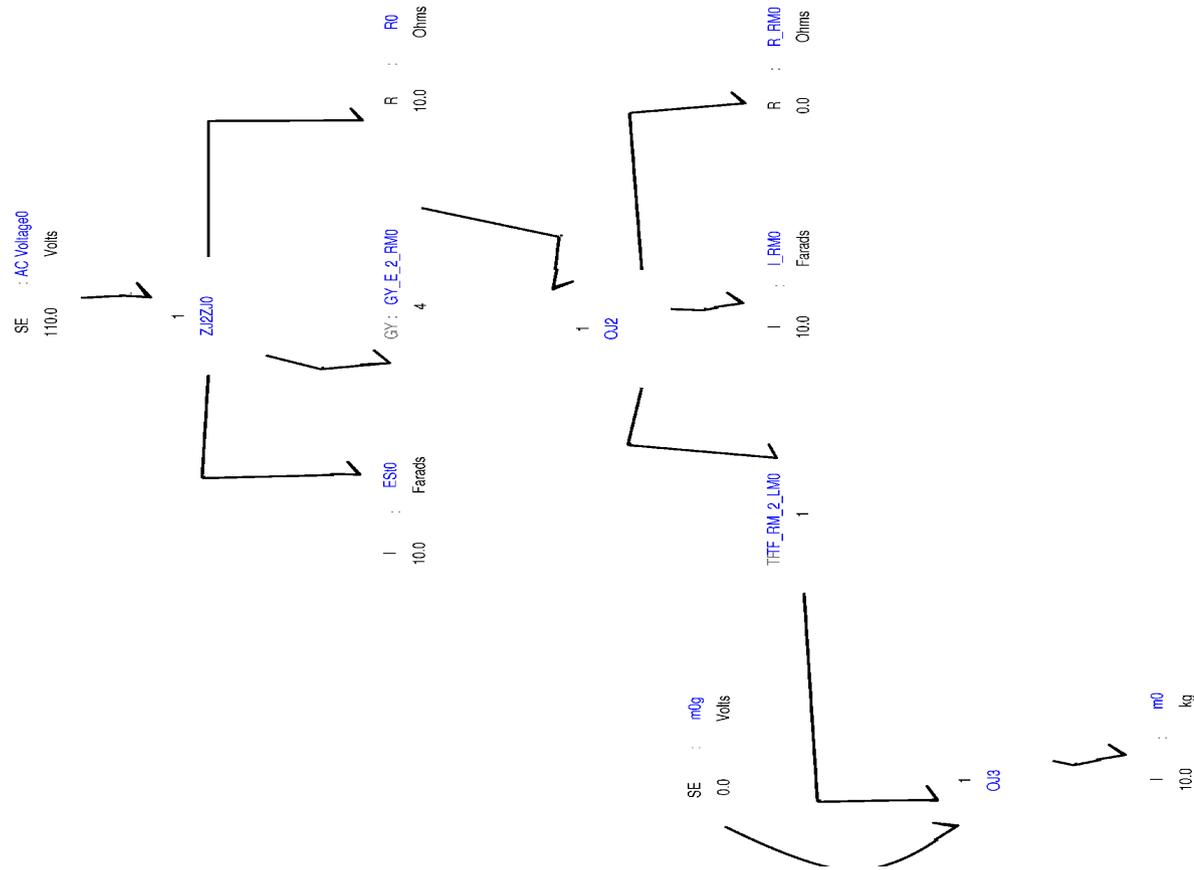
Generic link to connect two different formalisms (Here Real World Model and Idealized Physical Model)

Simple Example: Real World Model to Idealized Physical Model (RWM_2_IPM)



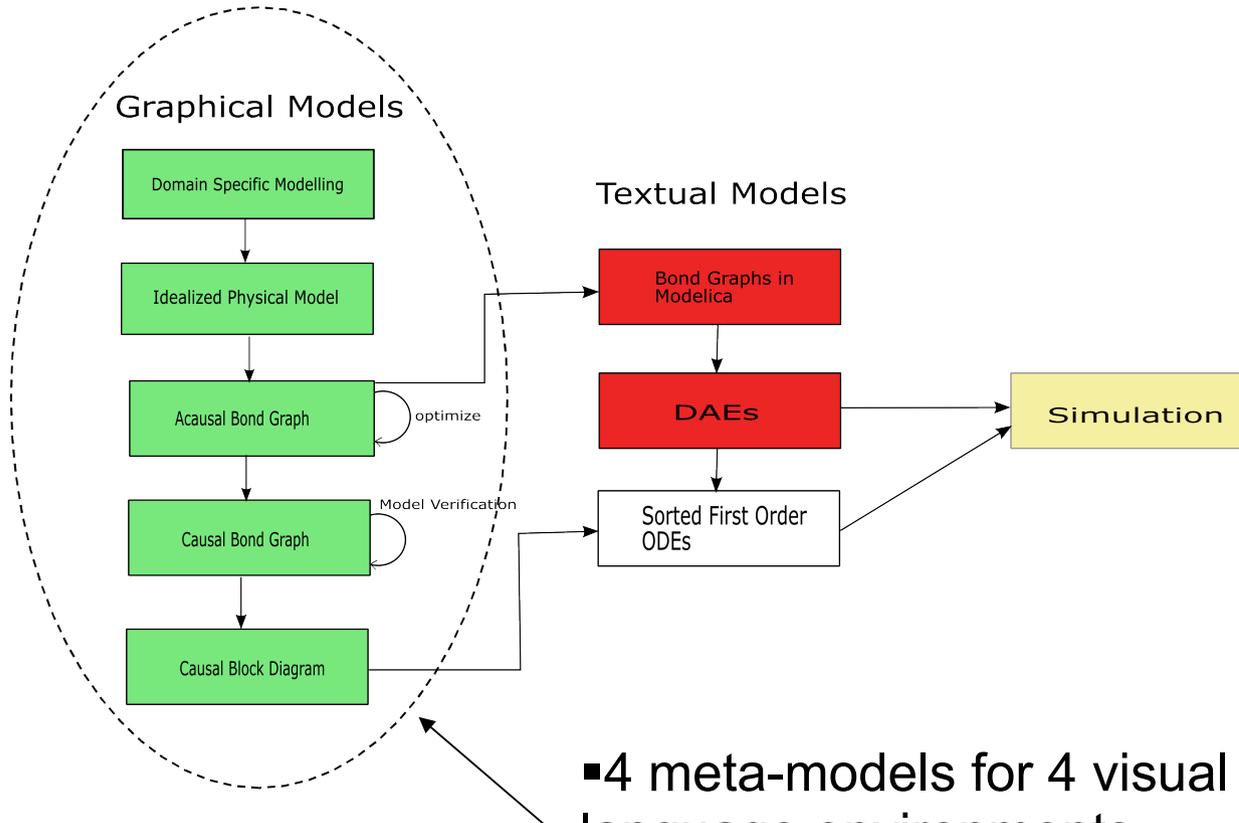
of Rules: 9
(5 were executed
in parallel)

More complex example: Idealized Physical Model to Acausal Bond Graph



Status: Final optimized acausal bond graph with hierarchical layout

Advantages of our approach



- 4 meta-models for 4 visual language environments
- Graph grammar for model transformation encoded in 114 rules

Approx. Lines of Code Generated

BondGraph : 6718 lines

IPM: 5601 lines

RWM: 1992 lines

CBD: 5464 lines

ABG_2_CBG: 11371 lines

CBG_2_CBD: 59568 lines

RWM_2_IPM: 2320 lines

IPM_2_ABG: 25381 lines

Total = 118451

Bug rate for new application development = 30 to several hundred bugs/KLOC (Average = 100 bugs/KLOC)

Fix rate = 2 mins /bug

Hours of work to fix bugs on an equivalent project = **394 hrs**

That may cost you (\$30/hr wage) = \$11820

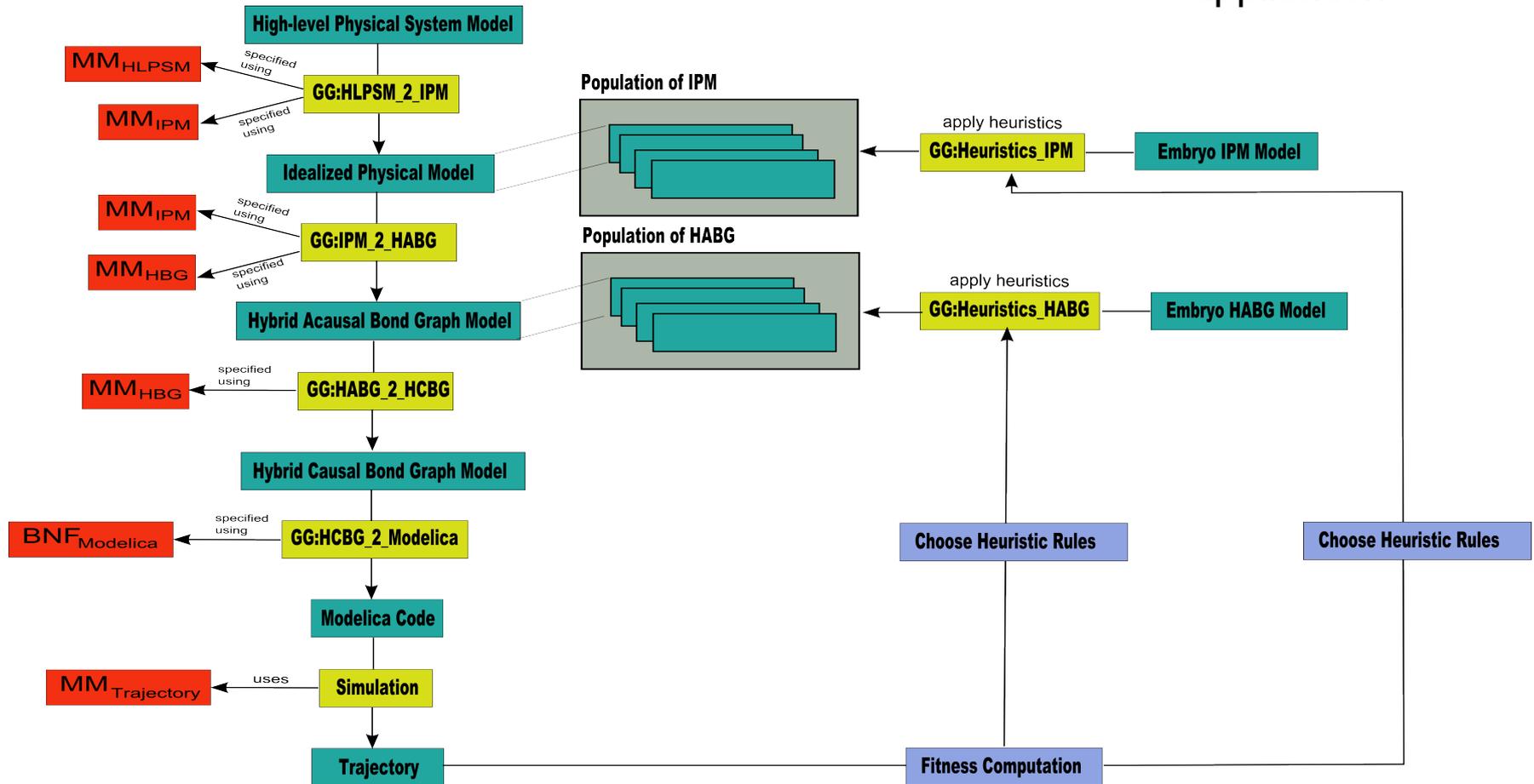
Design Space Exploration

Heuristics

Specified as
Graph Grammar Rules

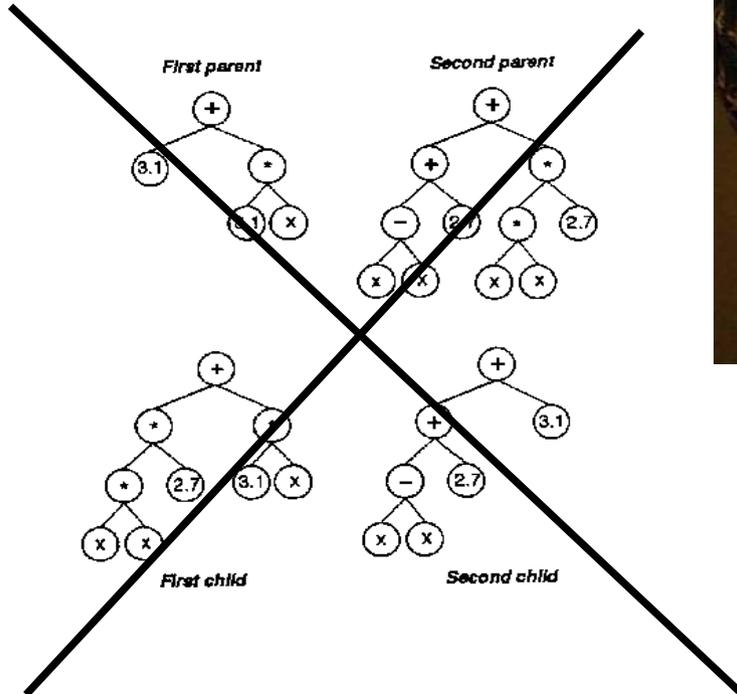
Search Method

Genetic Algorithm
on integer sequences
representing heuristic
application

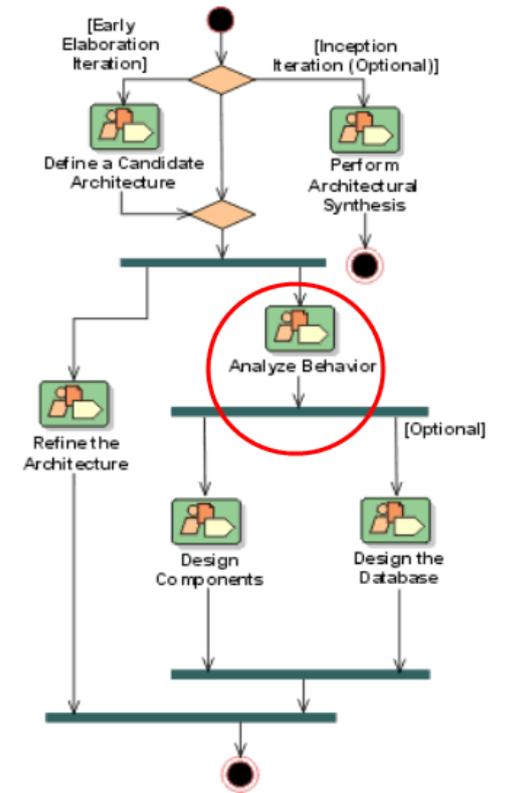
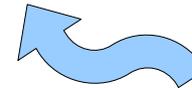


The goal: To show that domain specific knowledge drastically reduces design space size

Our vision for the future



Program = No



Model = Yes

Questions and Discussion