Exploring "Activity Tracking"

a Language Engineering perspective

Hans Vangheluwe


McGill

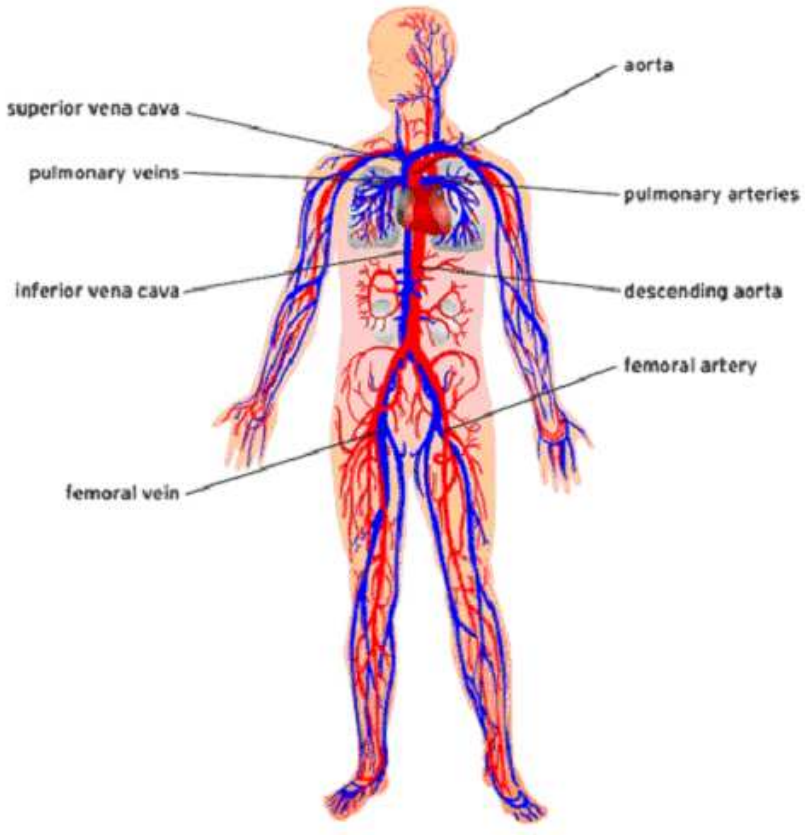
Modelling, Simulation and Design Lab (MSDL)
School of Computer Science, McGill University,
Montréal, Canada

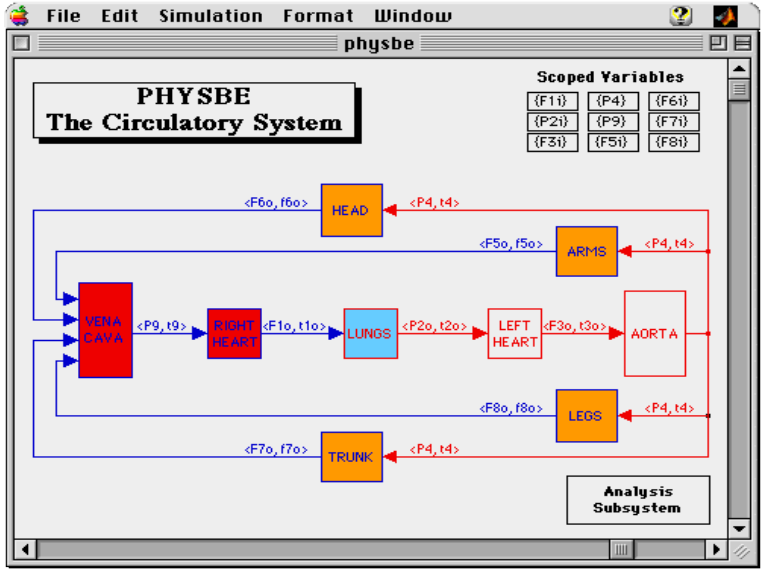
Interdisciplinar Colloquium on Activity-based Systems
21 April 2009

Cargèse, Corsica

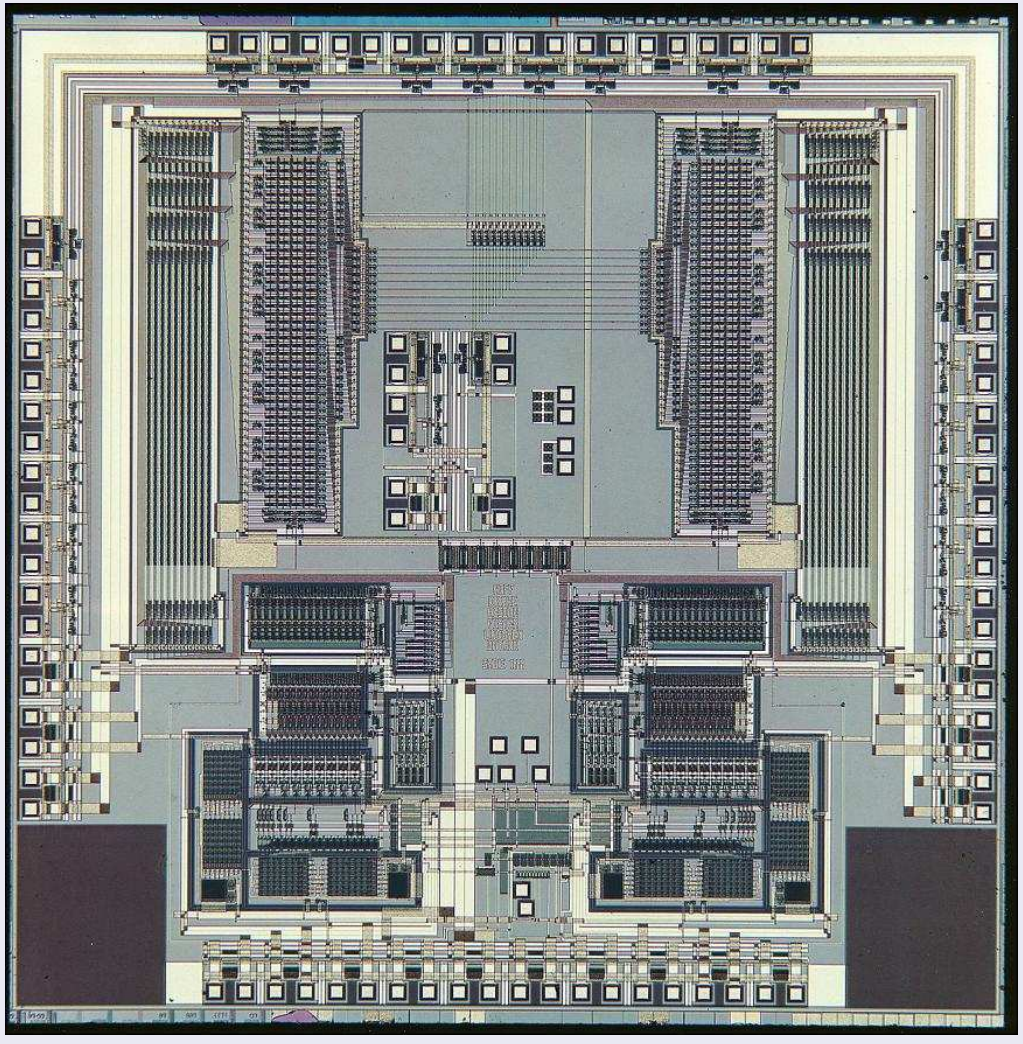
Dealing with Complexity

Different Abstraction Levels – properties preserved

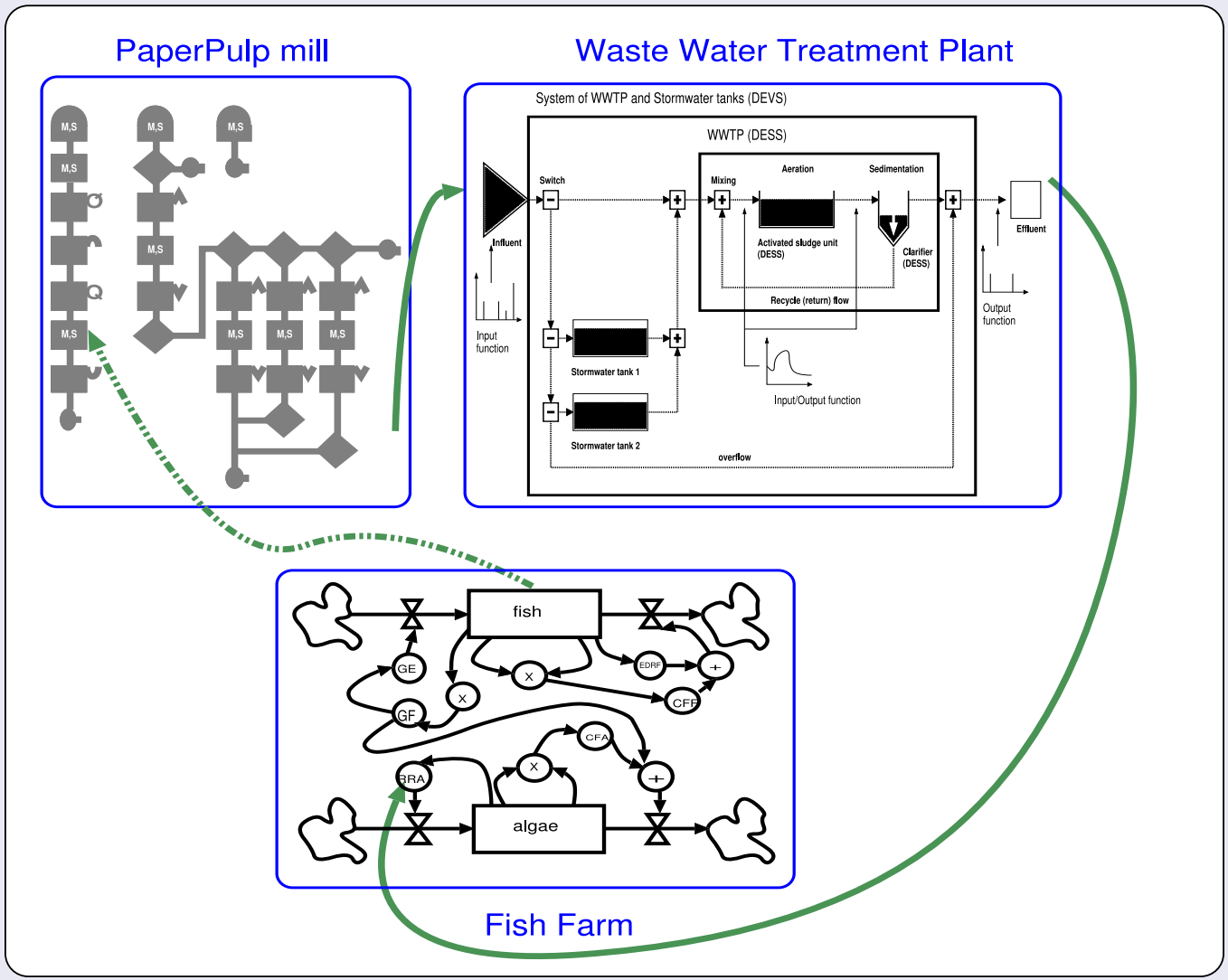




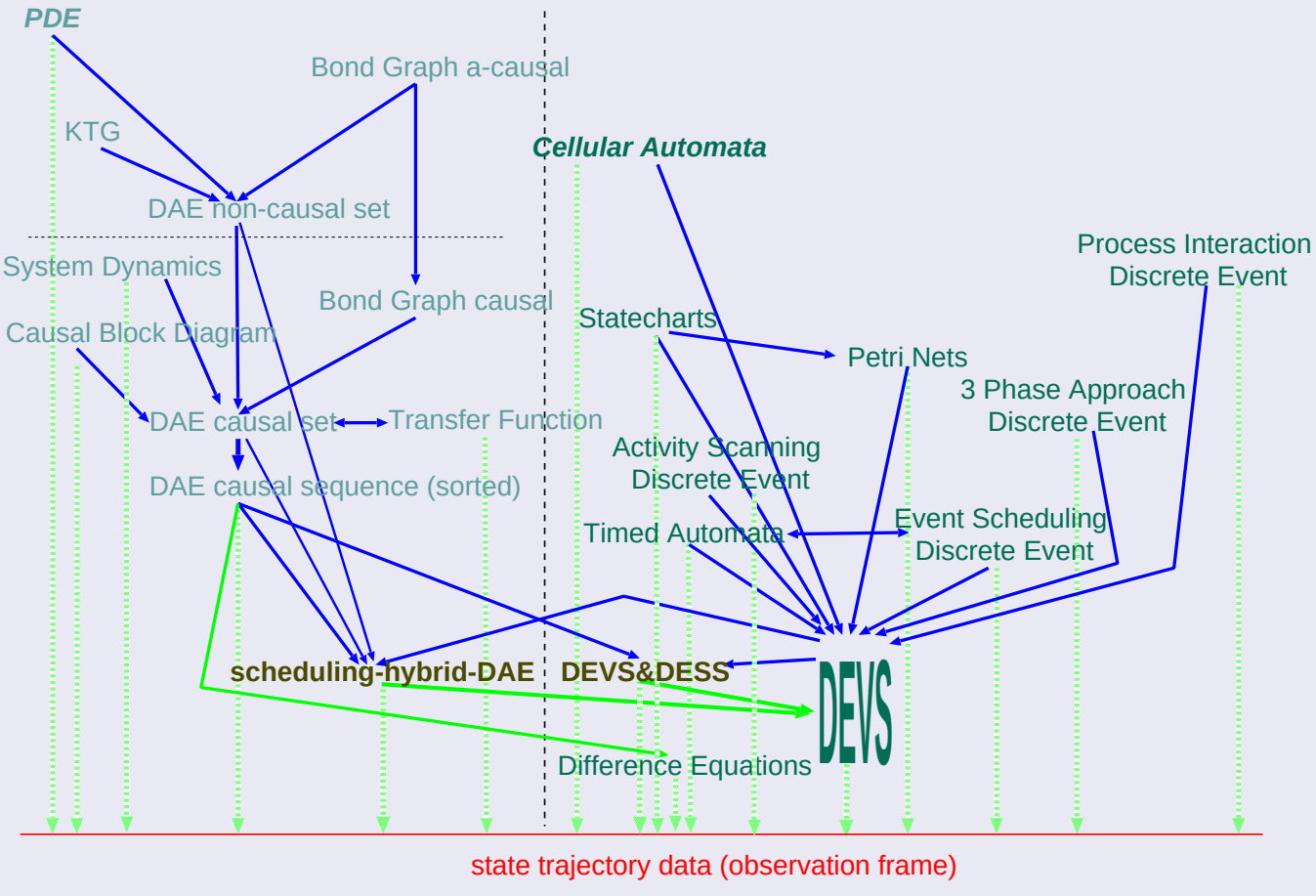
Number of Components – hierarchical (de-)composition



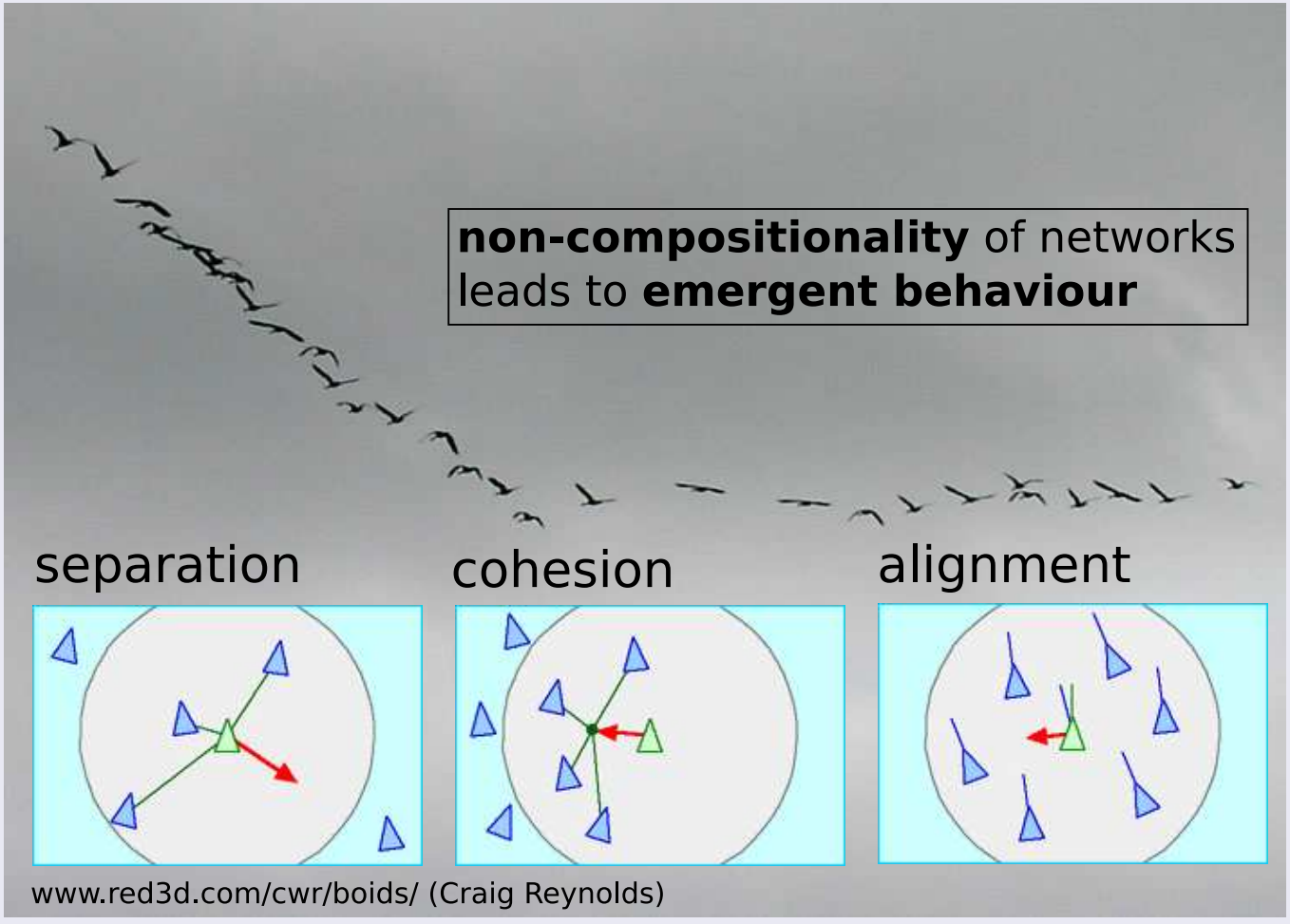
Components in Different Formalisms



Formalism Transformation Graph co-simulation vs. transformation



Non-compositional/Emergent Behaviour



Multi-Paradigm Modelling (minimize *accidental complexity*)

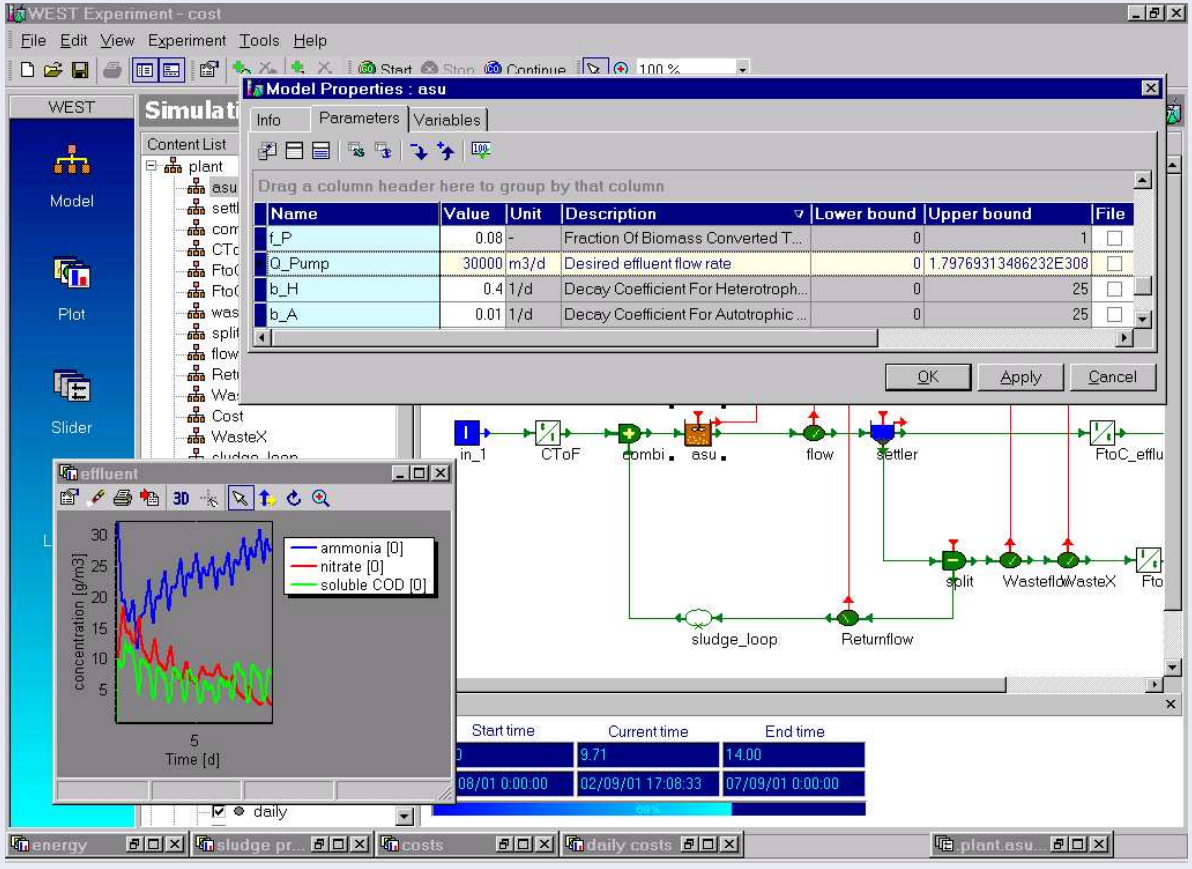
- at the most appropriate **level of abstraction**
- using the most appropriate **formalism(s)**
  Differential Algebraic Equations, Petri Nets, Bond Graphs, Statecharts, CSP, Queueing Networks, Lustre/Esterel, . . .
- with **transformations** as first-class models

Pieter J. Mosterman and Hans Vangheluwe.

Computer Automated Multi-Paradigm Modeling: An Introduction. Simulation 80(9):433–450, September 2004.

Special Issue: Grand Challenges for Modeling and Simulation.

WWTP Domain-Specific Modelling Environment



www.hemmis.com/products/west/

Henk Vanhooren, Jurgen Meirlaen, Youri Amerlinck, Filip Claeys, Hans Vangheluwe, and Peter A. Vanrolleghem.

WEST: Modelling biological wastewater treatment. Journal of Hydroinformatics, 5(1):27-50, 2003.

Why DS(V)M ? (as opposed to General Purpose modelling)

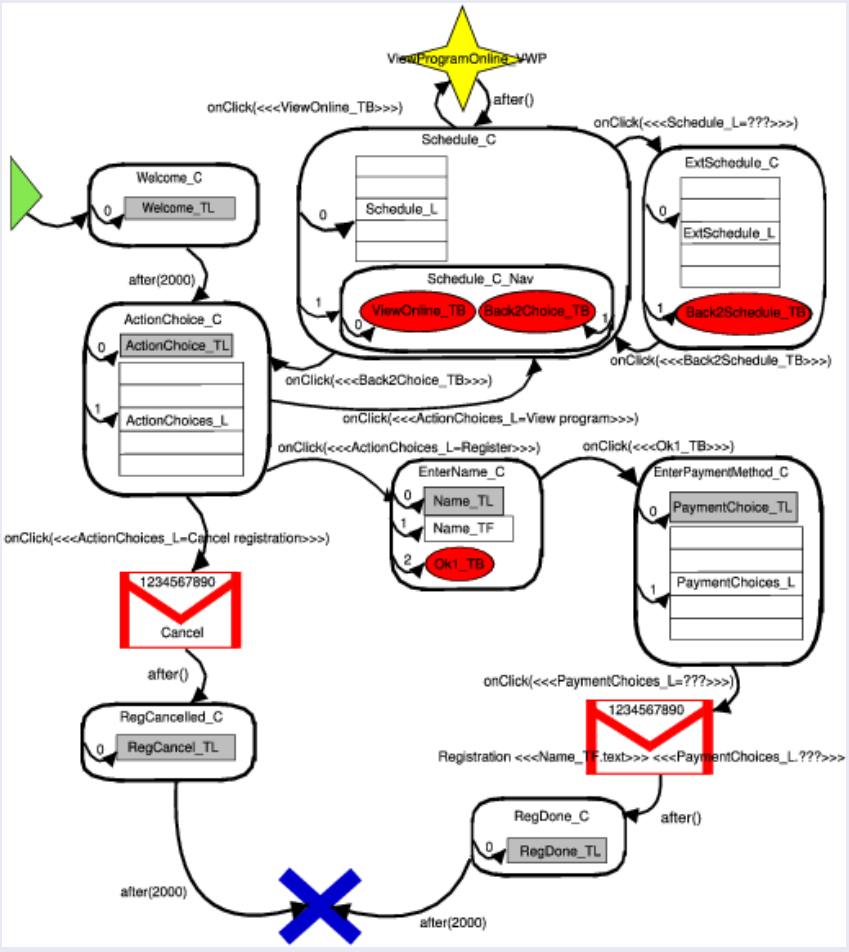
- **match the user's mental model** of the problem domain
- **maximally constrain** the user (to the problem at hand)
  $\Rightarrow$ easier to learn
  $\Rightarrow$ avoid errors
- **separate** domain-expert's work
  from analysis/transformation expert's work

Anecdotal evidence of 5 to 10 times speedup

Steven Kelly and Juha-Pekka Tolvanen.

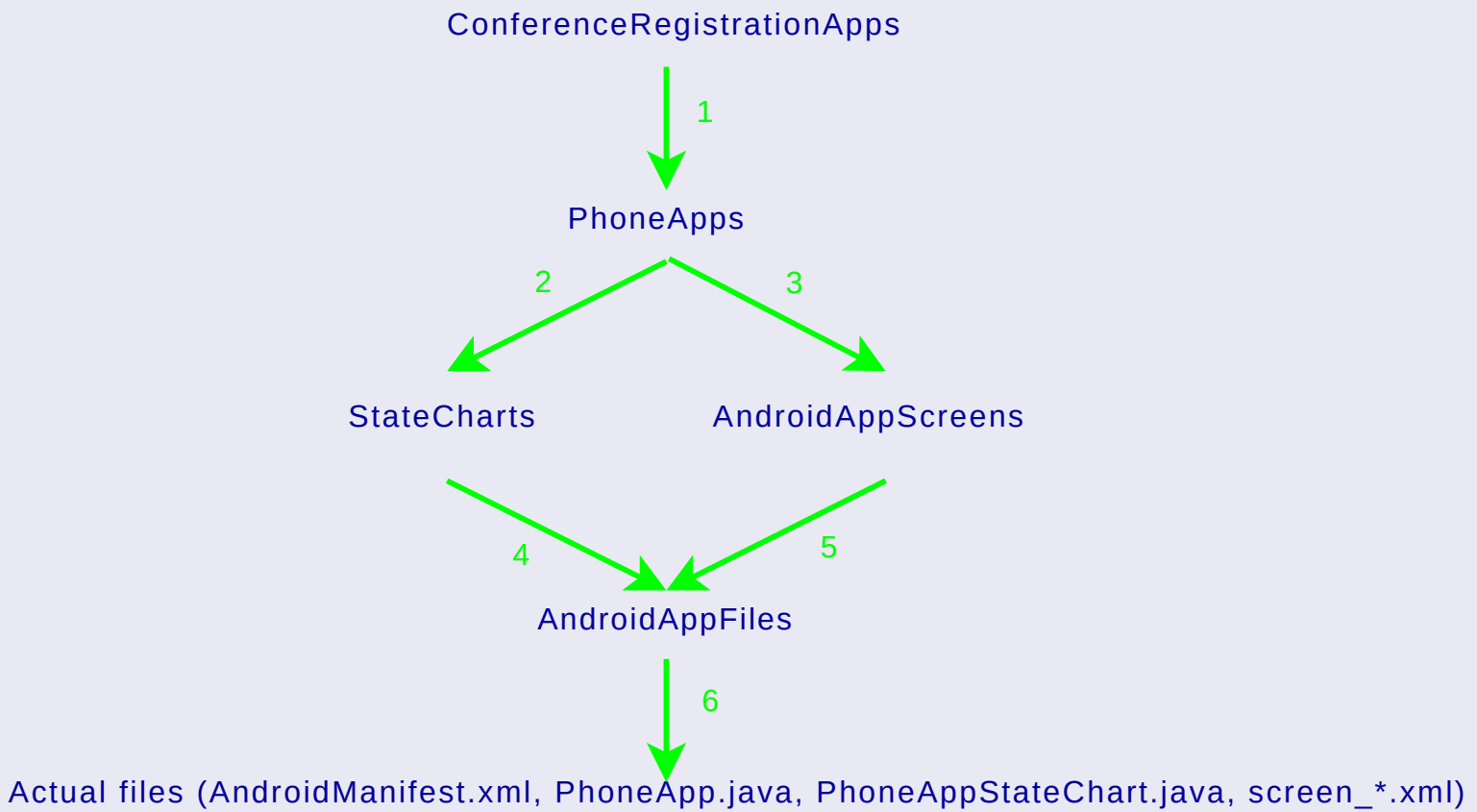
Domain-Specific Modeling: Enabling Full Code Generation. Wiley 2008.

DS(V)M example application, the PhoneApps Domain-Specific model

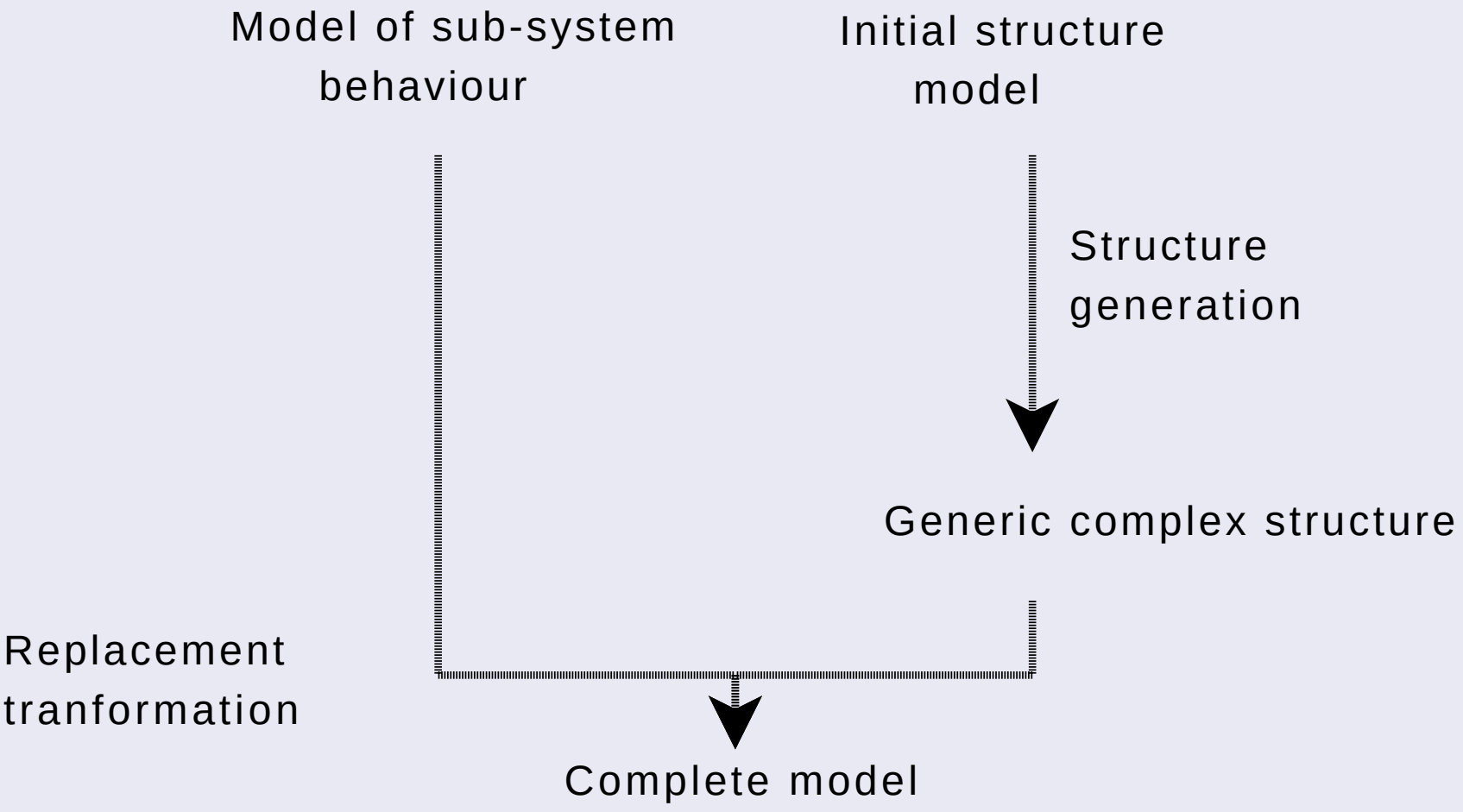


DS(V)M example application: conference registration (Google Android)

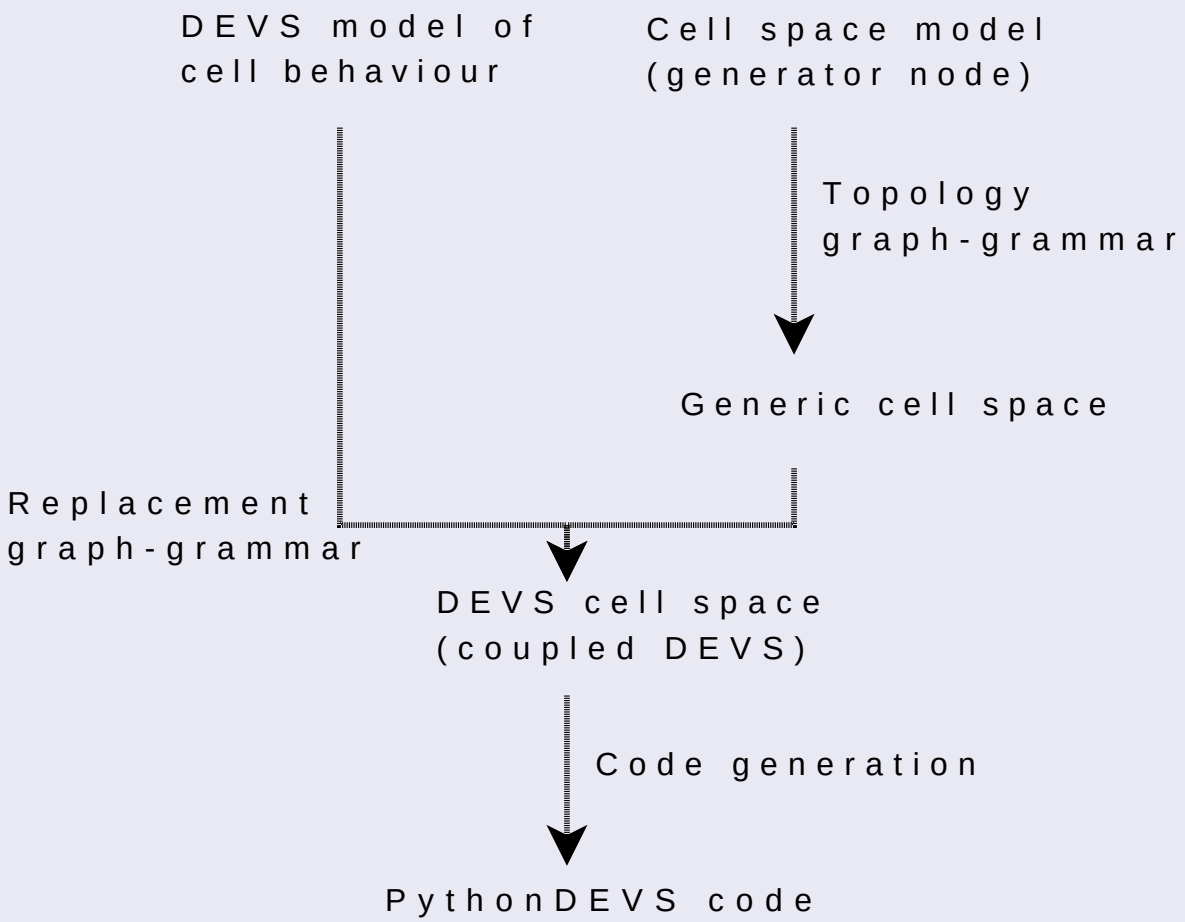
Only transform . . .



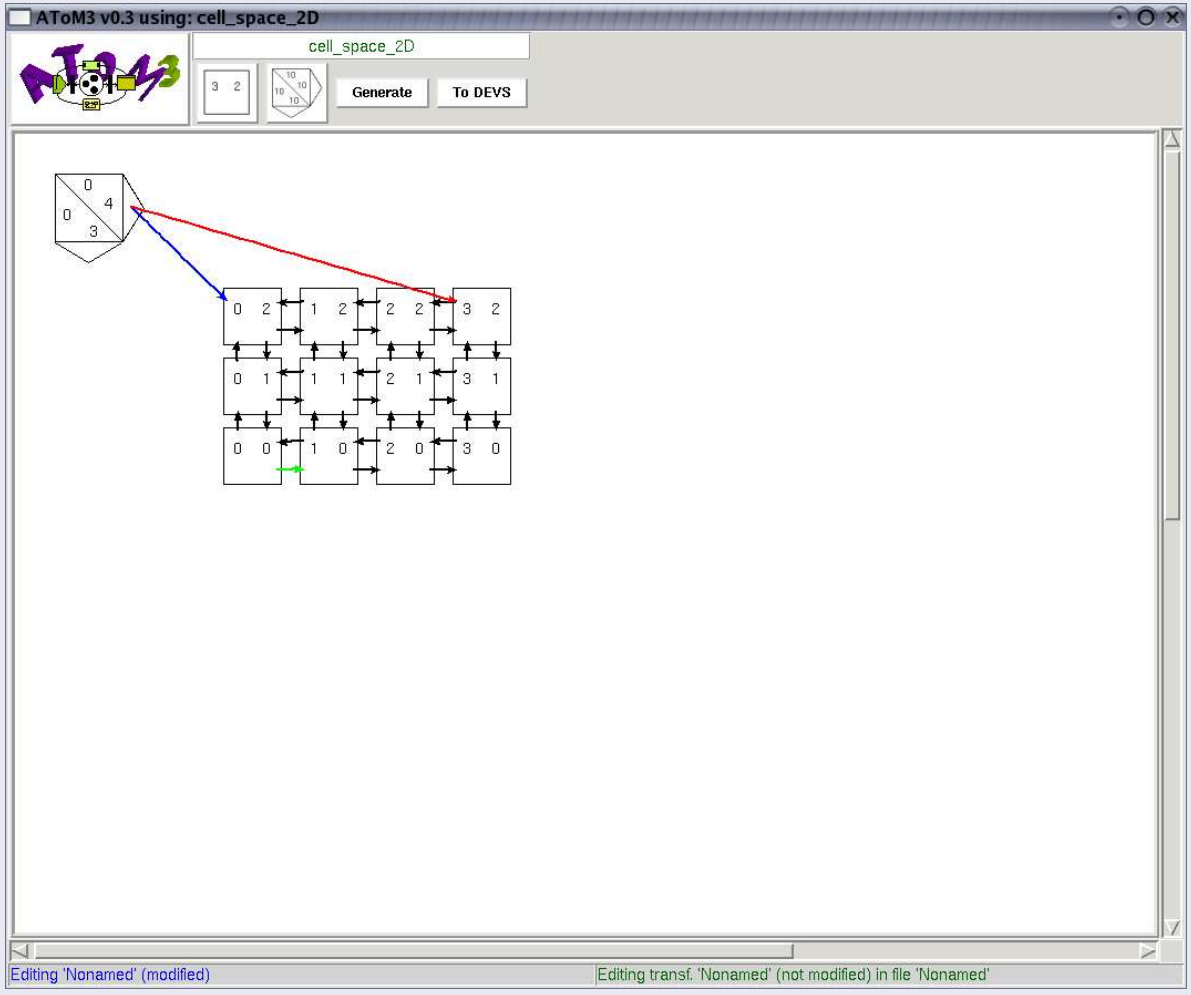
Weaving of Formalisms



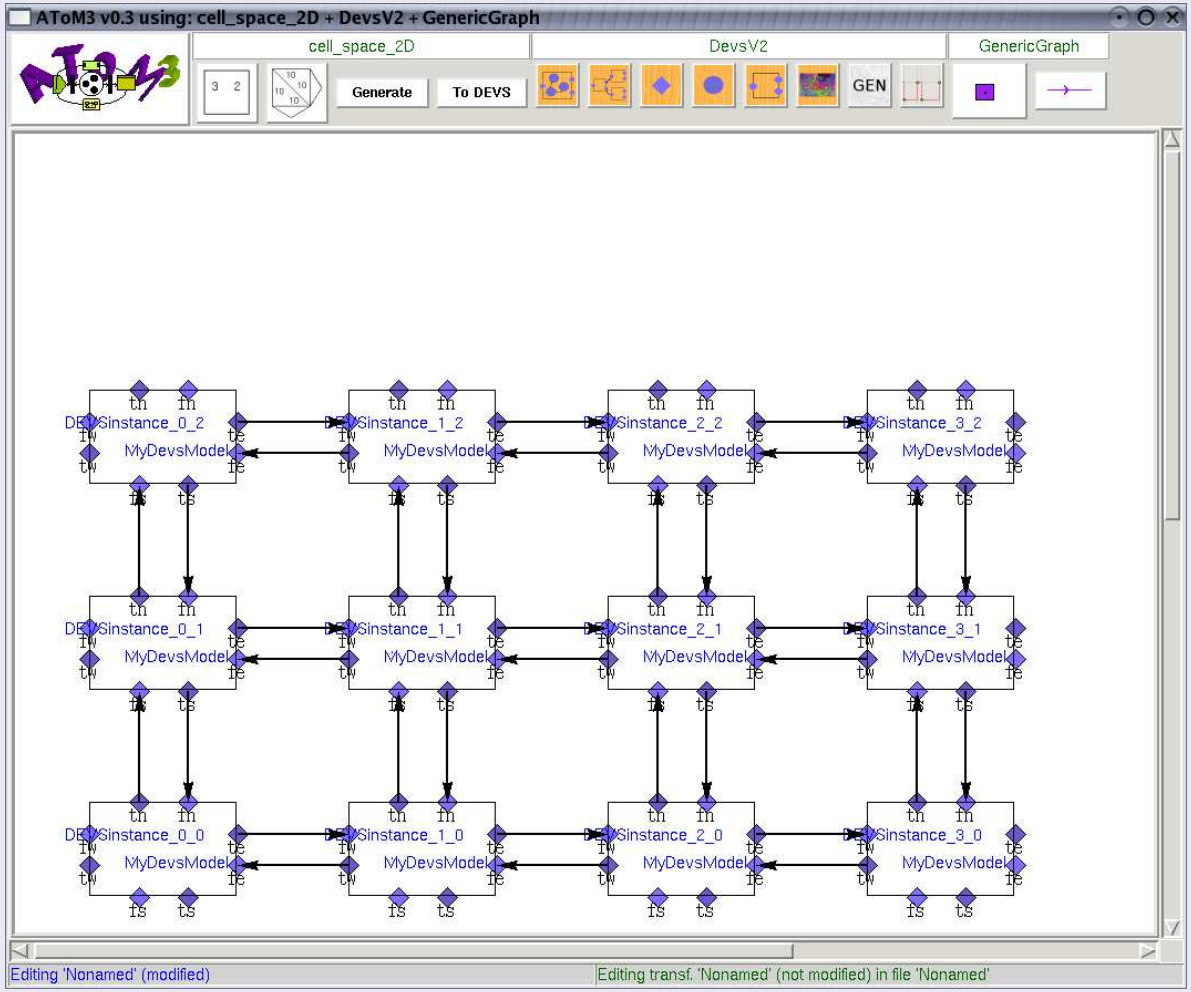
Concrete Example: DEVS + cellular structure



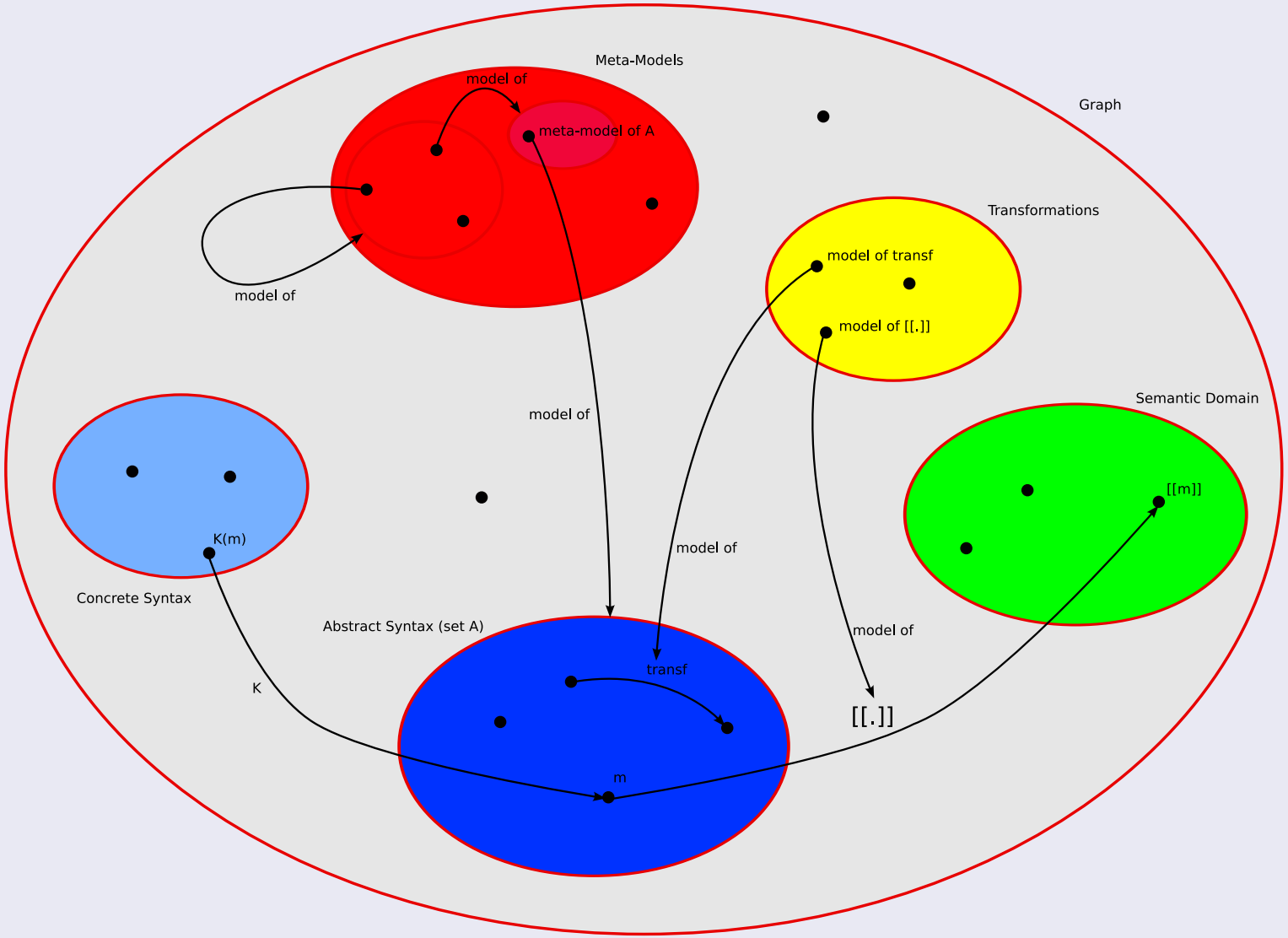
Cellspace generation



DEVS models (behaviour) woven in

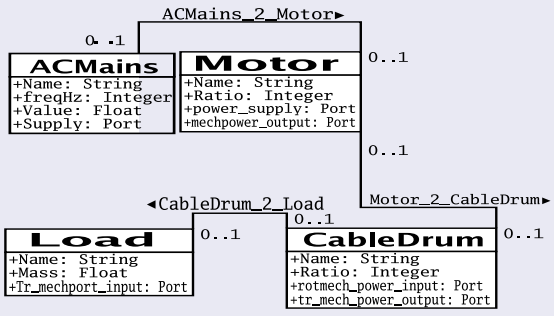


HOW TO do Modelling Language Engineering

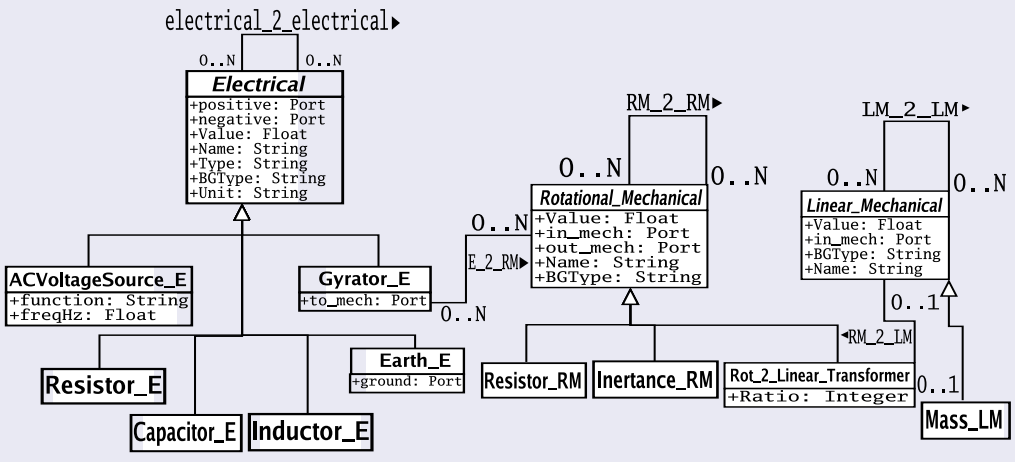


Metamodel

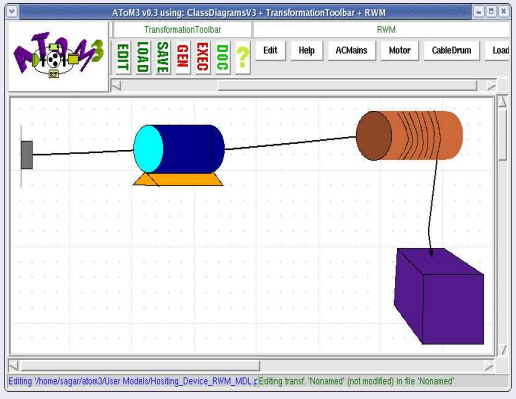
Real World Visual Modeling Formalism



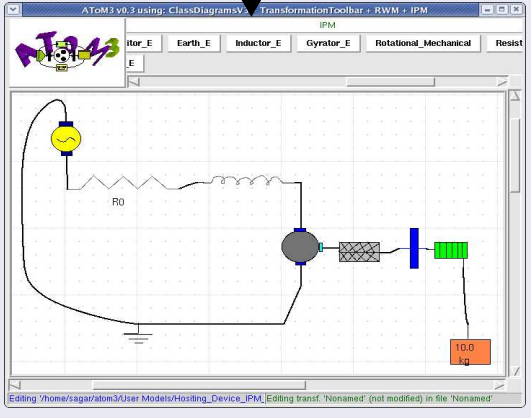
Idealized Physical Modeling Formalism



Generated Visual Modeling Environment

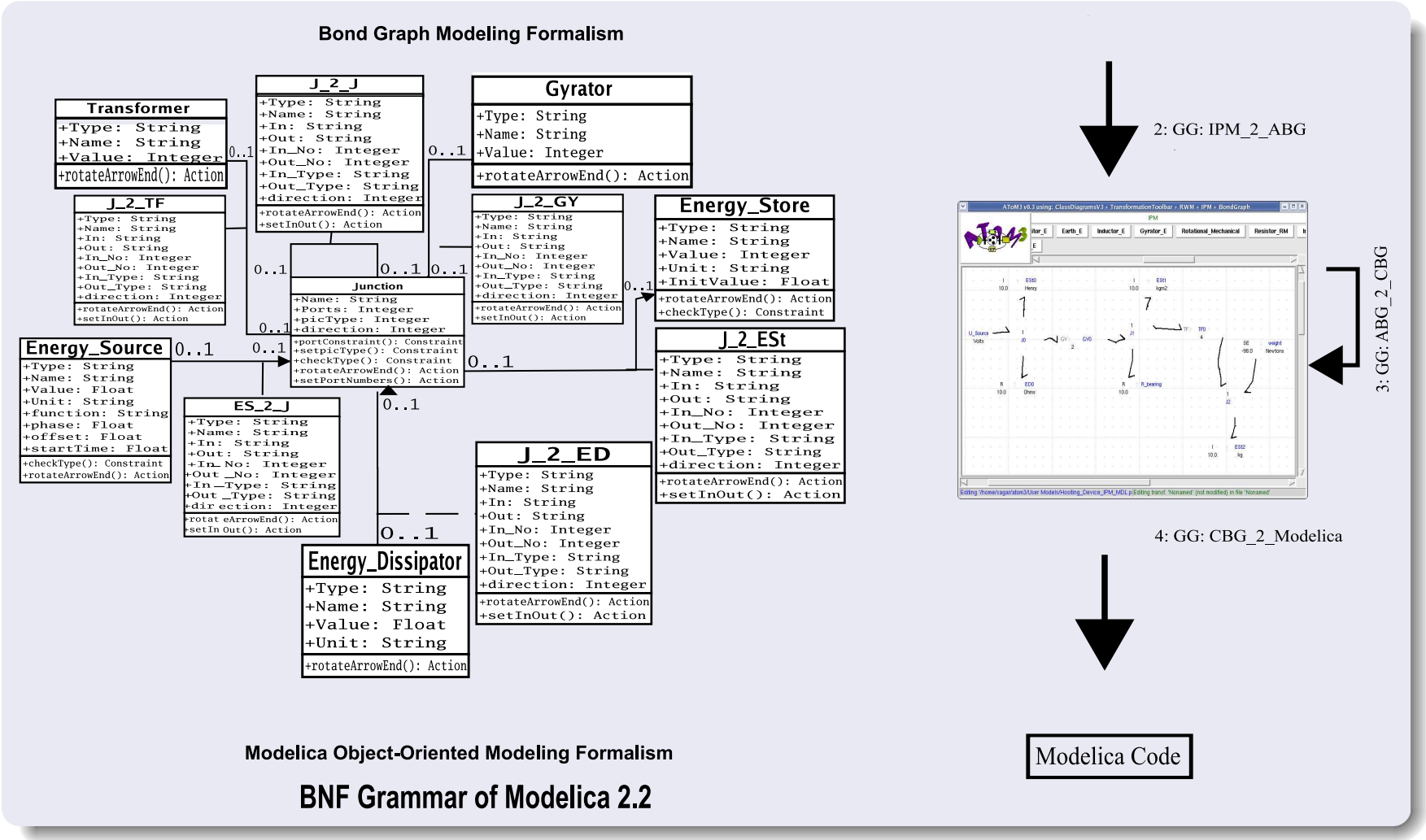


1: GG: RWVM_2_IPM



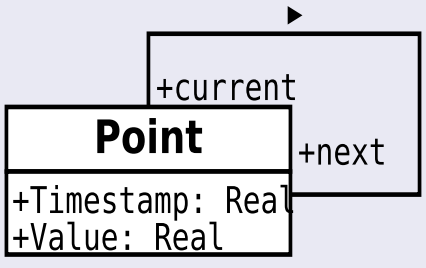
2: GG: IPM_2_ABG

Exploration

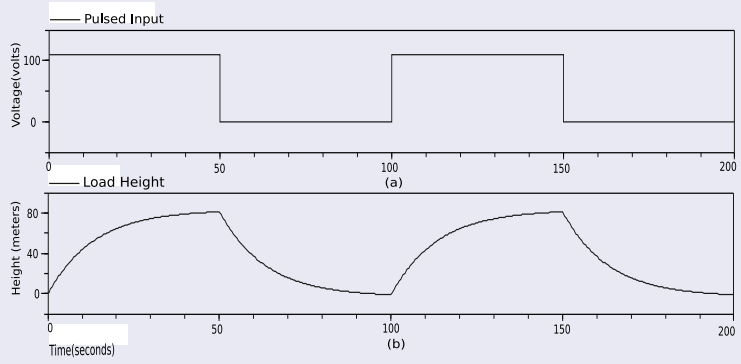


Exploration

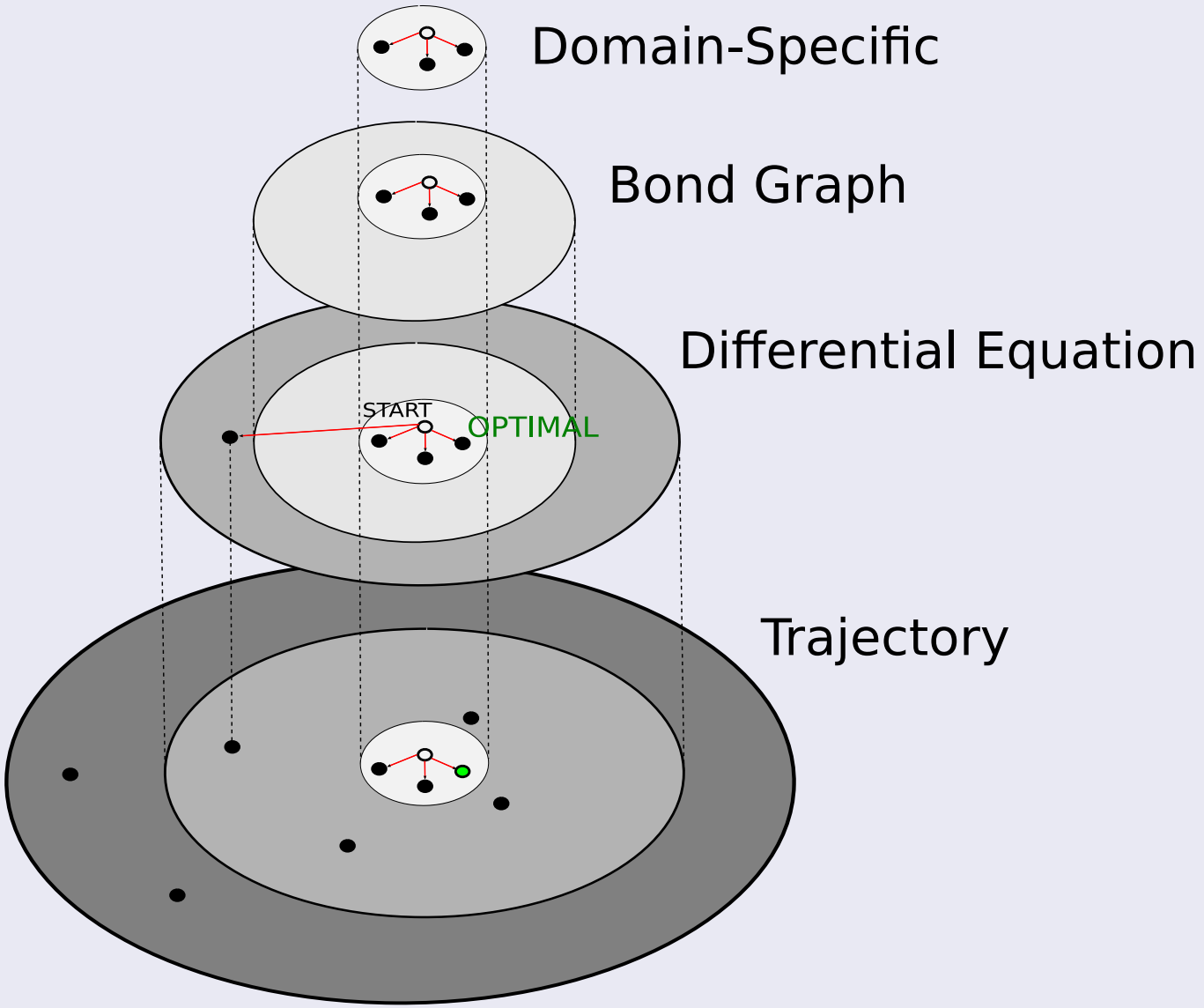
Trajectory Formalism



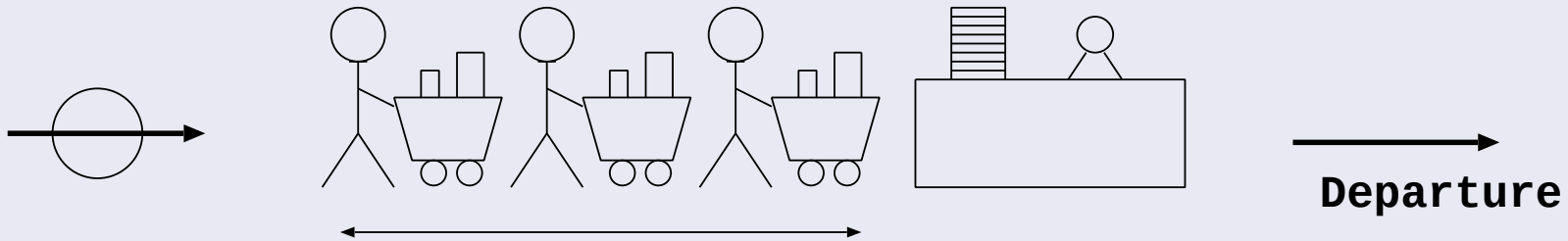
5: Simulation



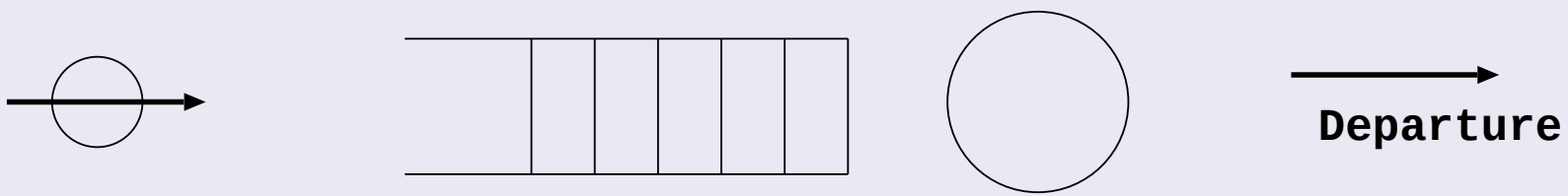
Exploring the Model Space



Activity Tracking? Single Server Queueing System

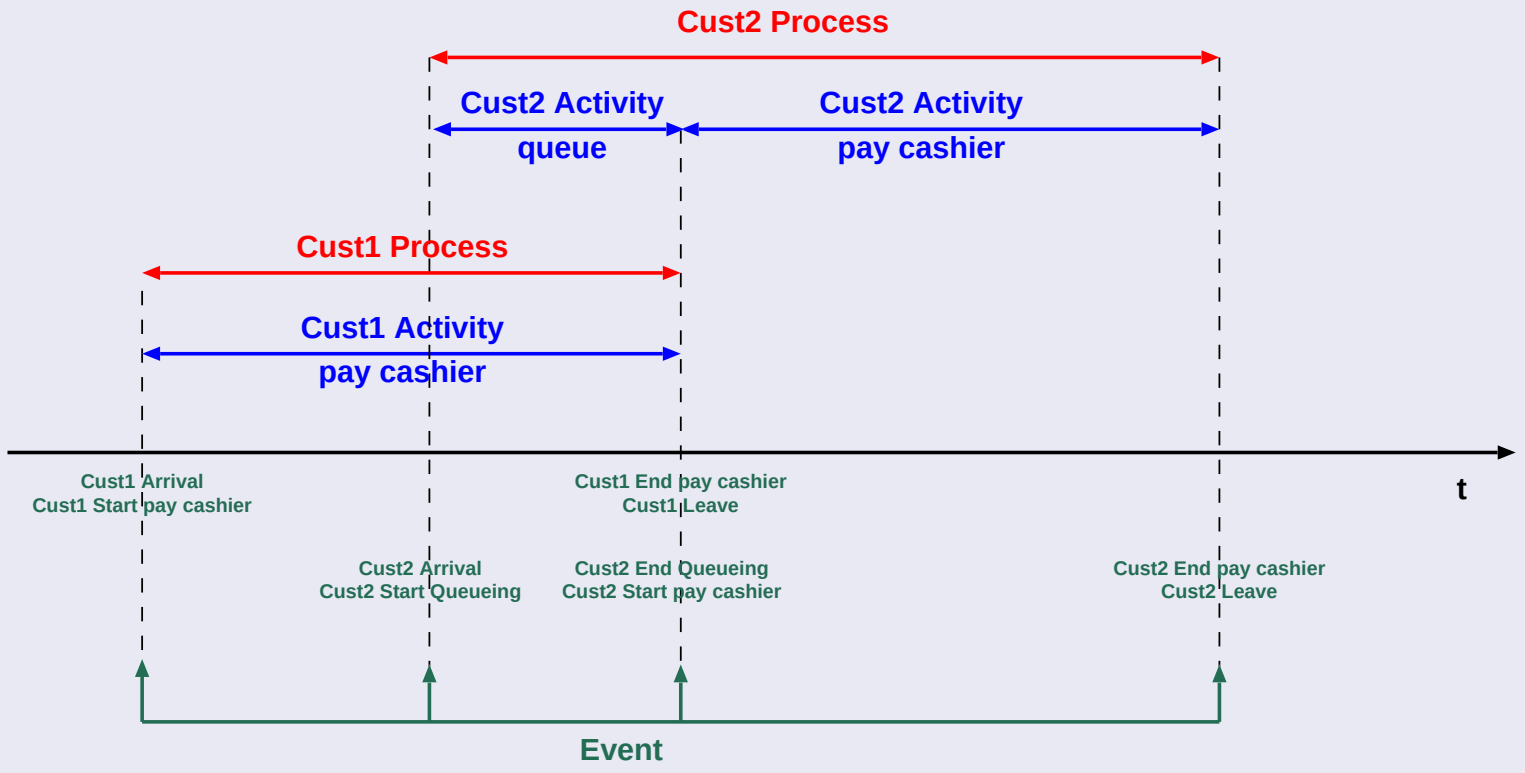


Physical View



Abstract View

World Views


Cust2 Process
Cust2 Activity
queue
Cust2 Activity
pay cashier
Cust1 Process
Cust1 Activity
pay cashier
t
Cust1 Arrival
Cust1 Start pay cashier
Cust1 End pay cashier
Cust1 Leave
Cust2 Arrival
Cust2 Start Queueing
Cust2 End Queueing
Cust2 Start pay cashier
Cust2 End pay cashier
Cust2 Leave
Event

Event Scheduling

- Identify *objects* and their *attributes*
- Identify *system attributes* (global)
- Define what causes *changes* in attribute value as *event*
- Write *event routine* for each event:
  - *modify state* (attributes)
  - *schedule* event(s) at $t + \Delta t, \Delta t \geq 0$
- Priorities for *tie-breaking*
- Event scheduling logic

Activity Scanning (rule-based)

Activity:

- condition: must be satisfied for activity to take place. Becomes true *only* at event times.
- actions: operations performed when condition becomes true

Time-advance mechanism:

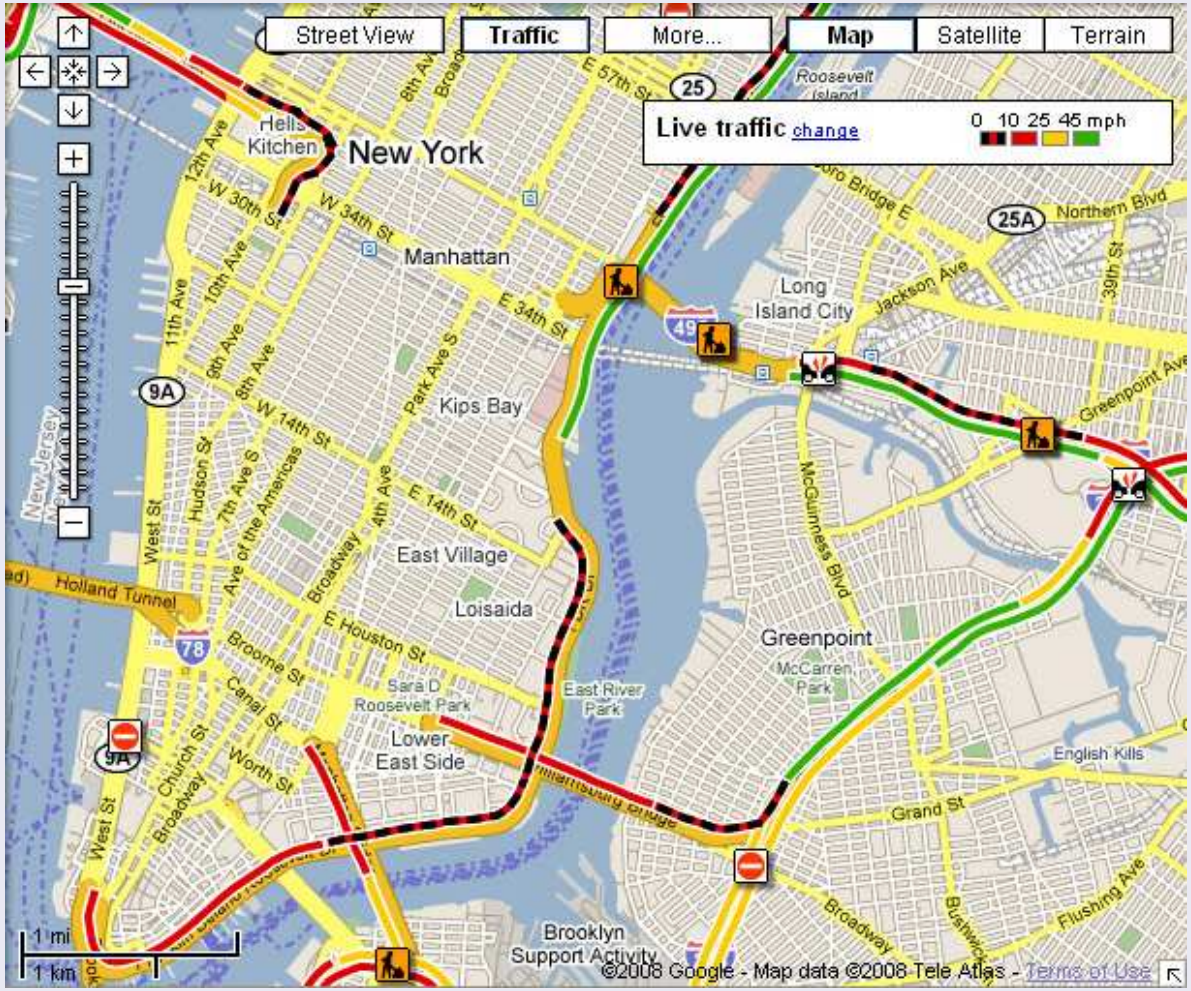
- fixed time-step

Also known as Two Phase Approach

Applications: Fire Spread

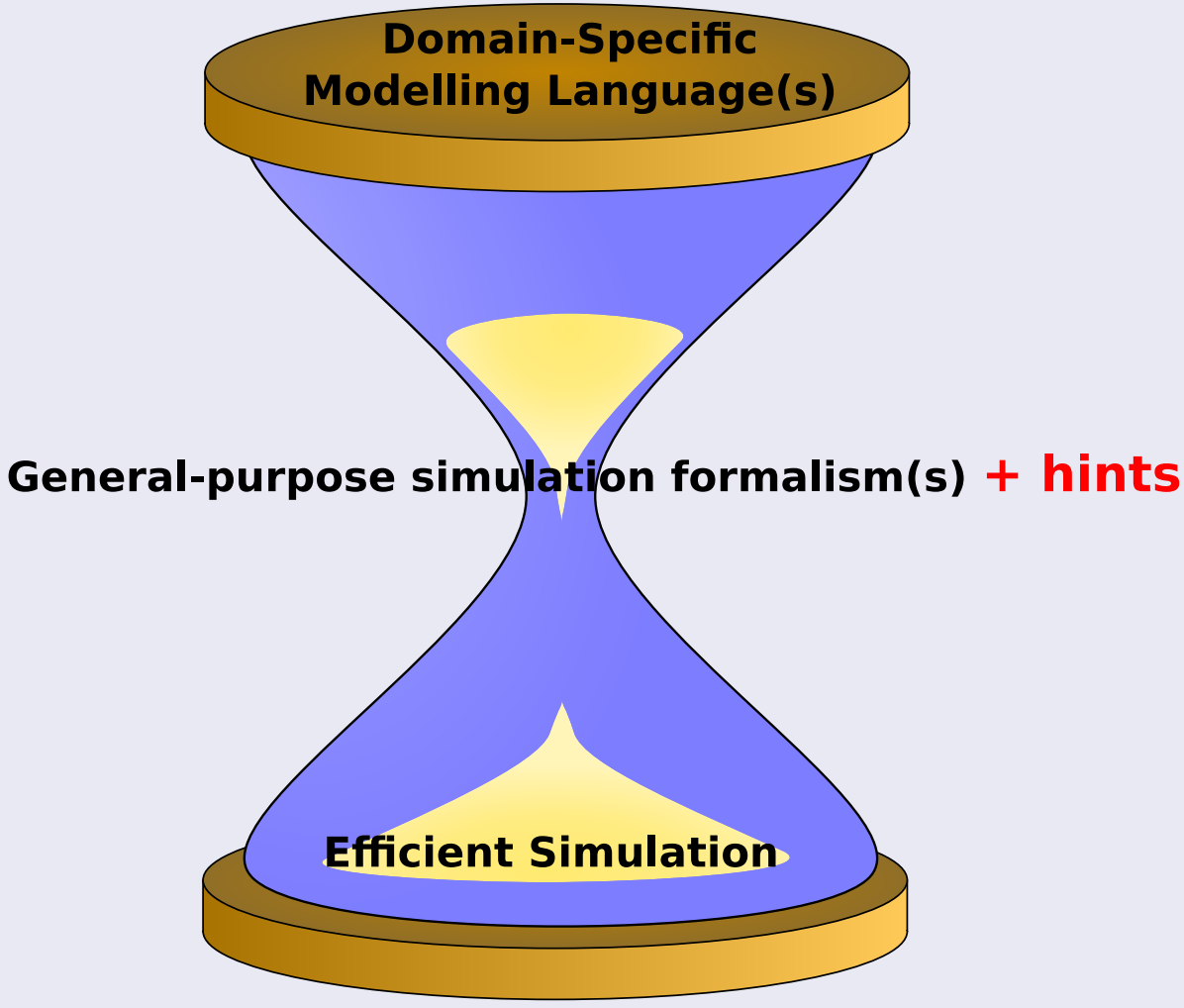
locality of activity in space and time

Domains: Population Dynamics



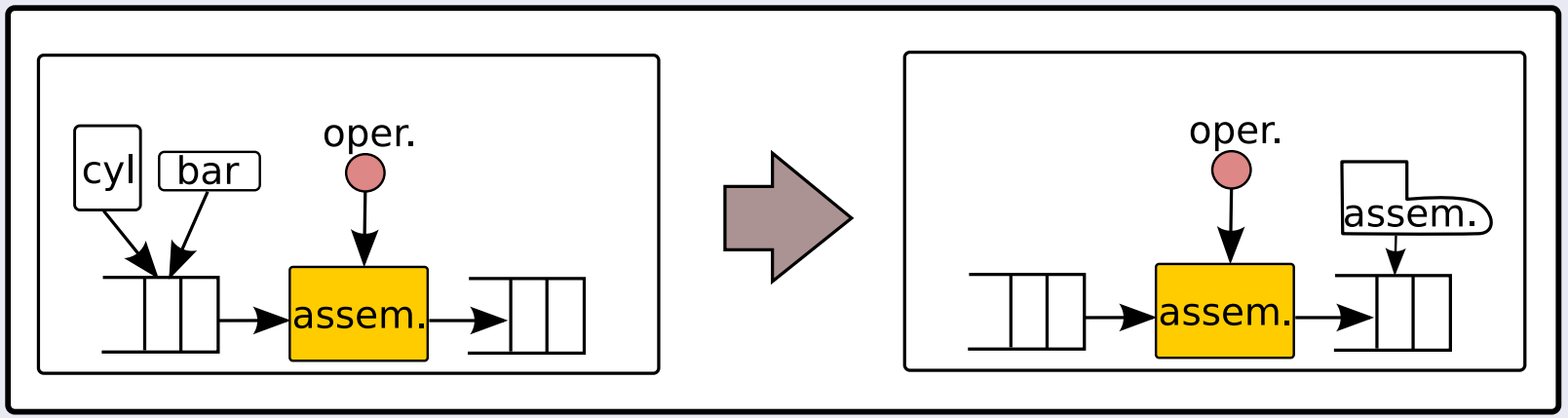
locality of activity in space and time

Having your cake and eat it . . .

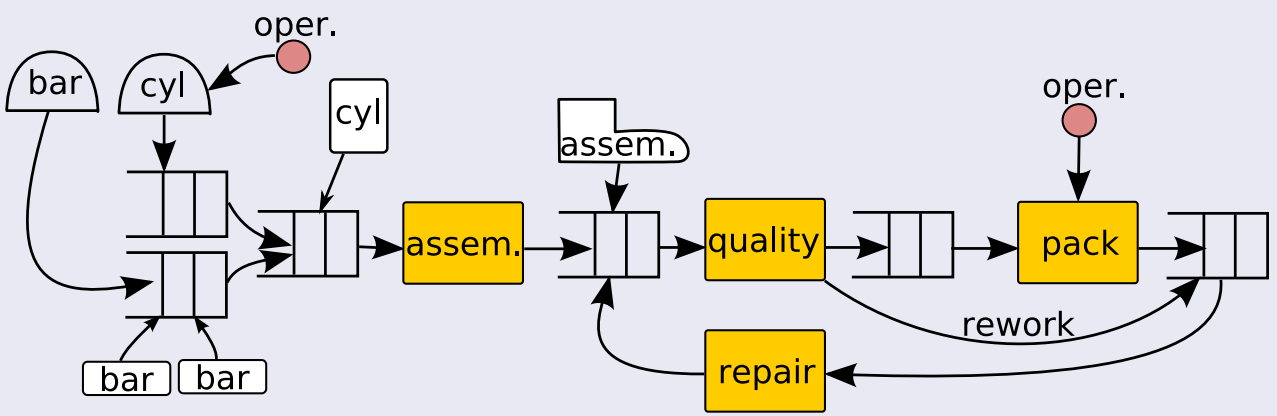


Use of Patterns/Rules to describe change

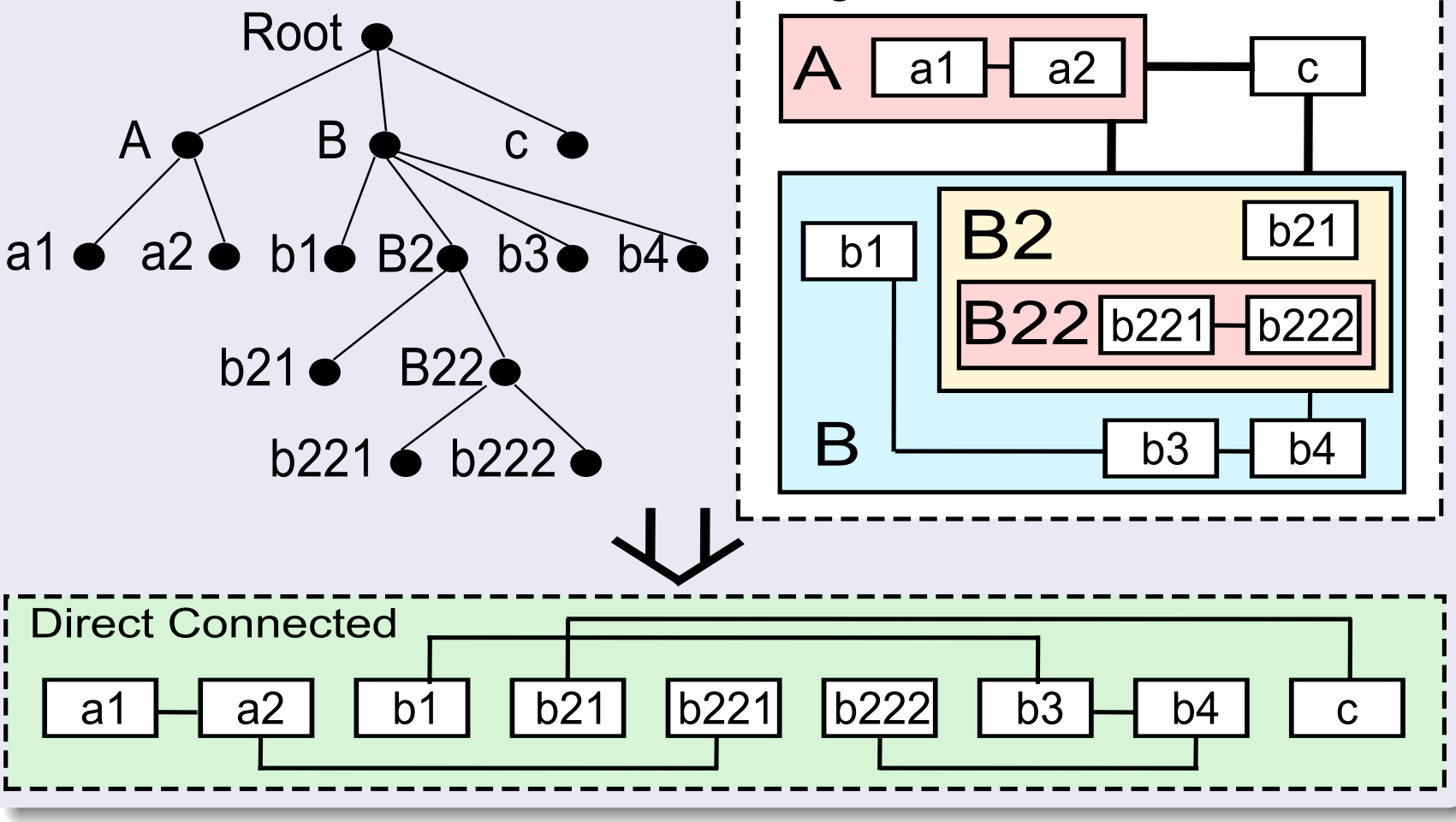
assemble



Note the use of **concrete** syntax !



Minimizing computation cost: Flattening



Conclusion: defining Activity Scanning

- from the modelling point of view
- from the simulation (computation) point of view
  spatio-temporal heterogeneity/locality
- language engineering viewpoint?

“Activity Tracking” Google search . . .