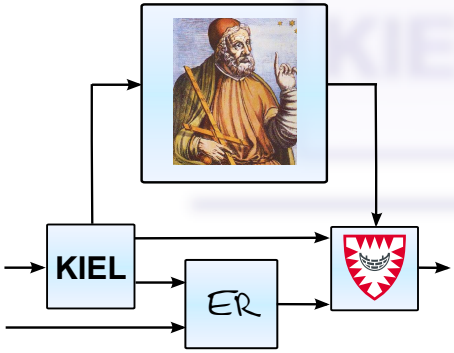
# Semantics and Executions of Domain Specific Models

## KlePto and an Execution Infrastructure

Christian Motika

Real-Time Systems and Embedded Systems Group
Department of Computer Science
Christian-Albrechts-Universität zu Kiel, Germany




MEMWe Workshop 2010
Leipzig, 30.09.2010

2010-10-03

# Semantics and Executions of Domain Specific Models

Semantics and Executions of Domain Specific Models
KlePto and an Execution Infrastructure

Christian Motika

Real-Time Systems and Embedded Systems Group
Department of Computer Science
Christian-Albrechts-Universität zu Kiel, Germany


MEMWe Workshop 2010
Leipzig, 30.09.2010

# Motivation

- KIELER „...is a research project about enhancing the graphical model-based system design of complex systems.“
- Diagram creation, editing and browsing, automatic layout, ...
- Also desired: Simulation runs
- **Idea:** Swap out simulation computation
- **Solution proposed:** Usage of Ptolemy as a *simulation engine*
- „**K**IELER **le**vering **Pto**lemy“ $\longrightarrow$ KlePto

# Aims - Model semantics

1. Simplicity: Avoid new language
2. Flexibility: Avoid concrete language
3. Usability: Common technologies
4. Abstraction: High level

# Aims - Model execution

1. Simplicity: Interface
2. Flexibility: Data representations
3. Usability: Simple user interface
4. Interactivity: Discrete, stepwise executions
5. Performance: Concurrency

# Overview

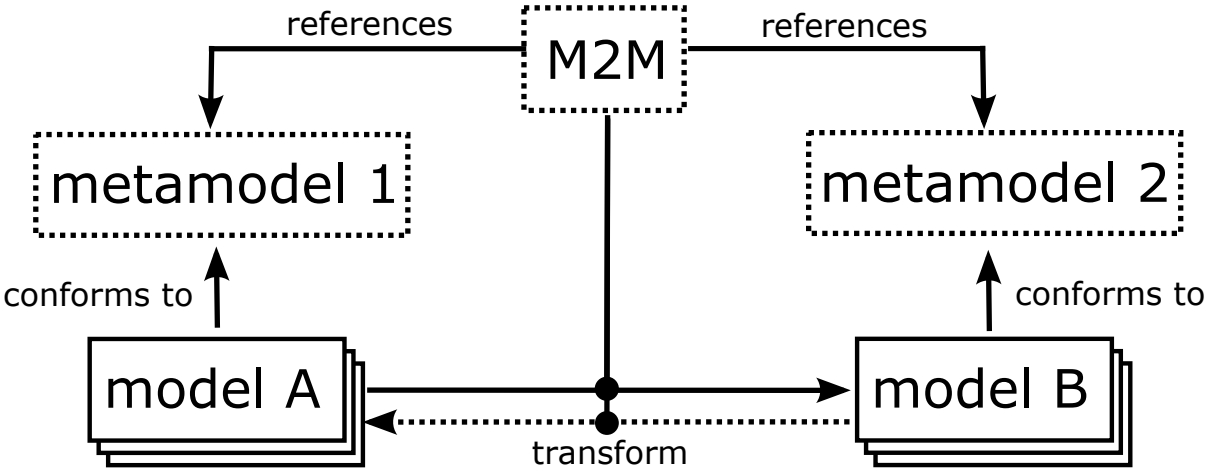
- Model Semantics
  - Specification Options
  - Ptolemy
  - KlePto Concept
  - Case Study: SyncCharts
- Model Execution
  - Framework Overview
  - User Interface
- Summary

# DSL Semantics

- Different approaches:
  1. Operational
     - Action language → Model Execution Framework (MXF)
     - Extend metamodel
     - Additional meaning must be defined (formal or informal)
  2. Denotational
     - Semantic domain
     - Model transformation
- Denotational approach:
  - ☺ No new language
  - ☺ Direct relation
  - ☺ Abstraction level

# Model Transformations

- Key role in MDSD
- Types:
  1. Model to text transformations (M2T)
  2. Model to model transformations (M2M)
  3. Inplace transformations
- M2M:



# Used Technology

- Eclipse Framework (plug-in based)
- Metamodeling:
  - Eclipse Modeling Framework (EMF)
  - Ecore
- Model Transformations:
  - Xpand, Xtend
- DSL Editors:
  - Graphical Modeling Framework (GMF)
  - Graphical Editing Framework (GEF)
- Ptolemy

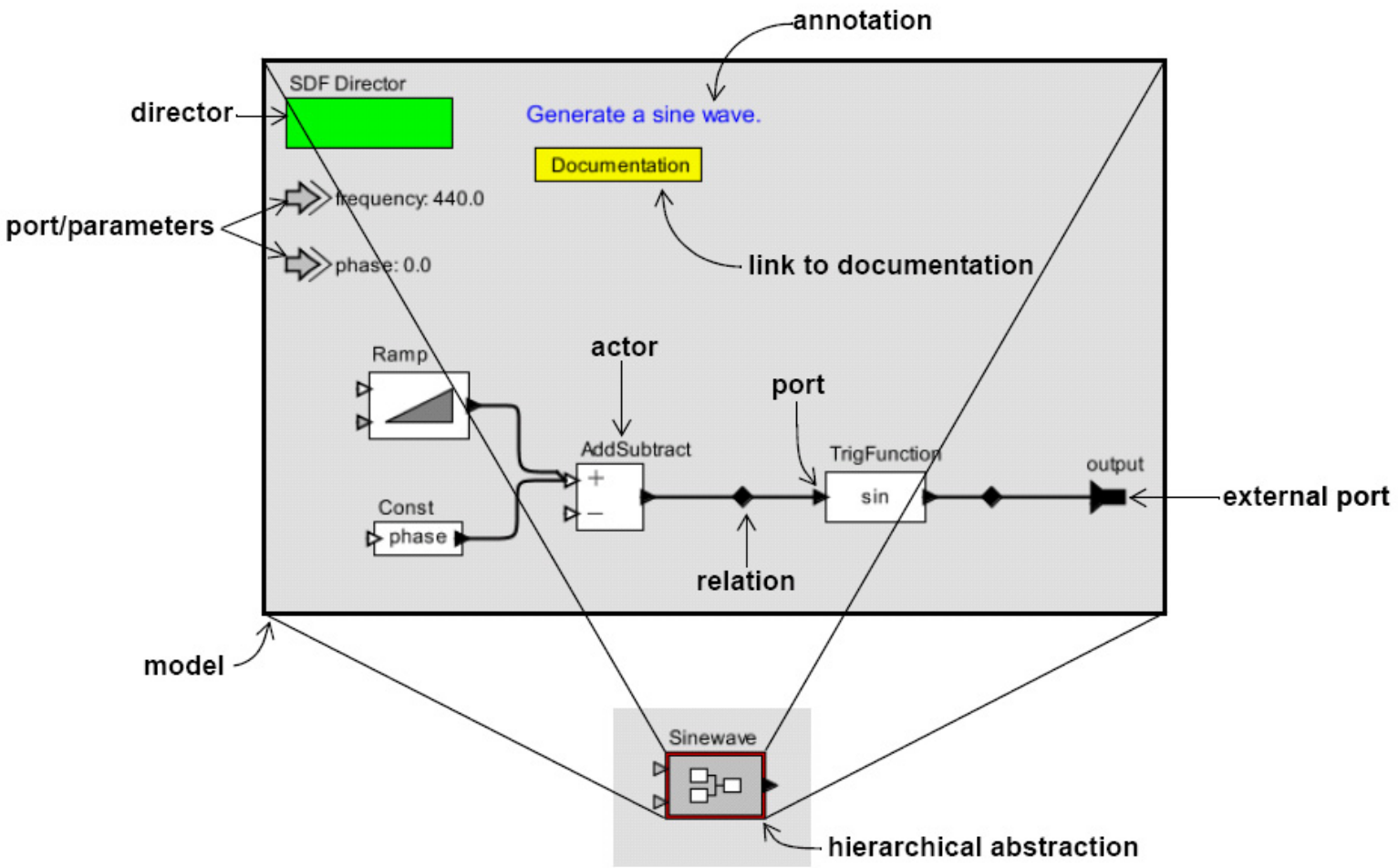
# Ptolemy

- „The Ptolemy project studies heterogeneous modeling, simulation, and design of concurrent systems.“

  Introduction to Ptolemy II, UC Berkeley

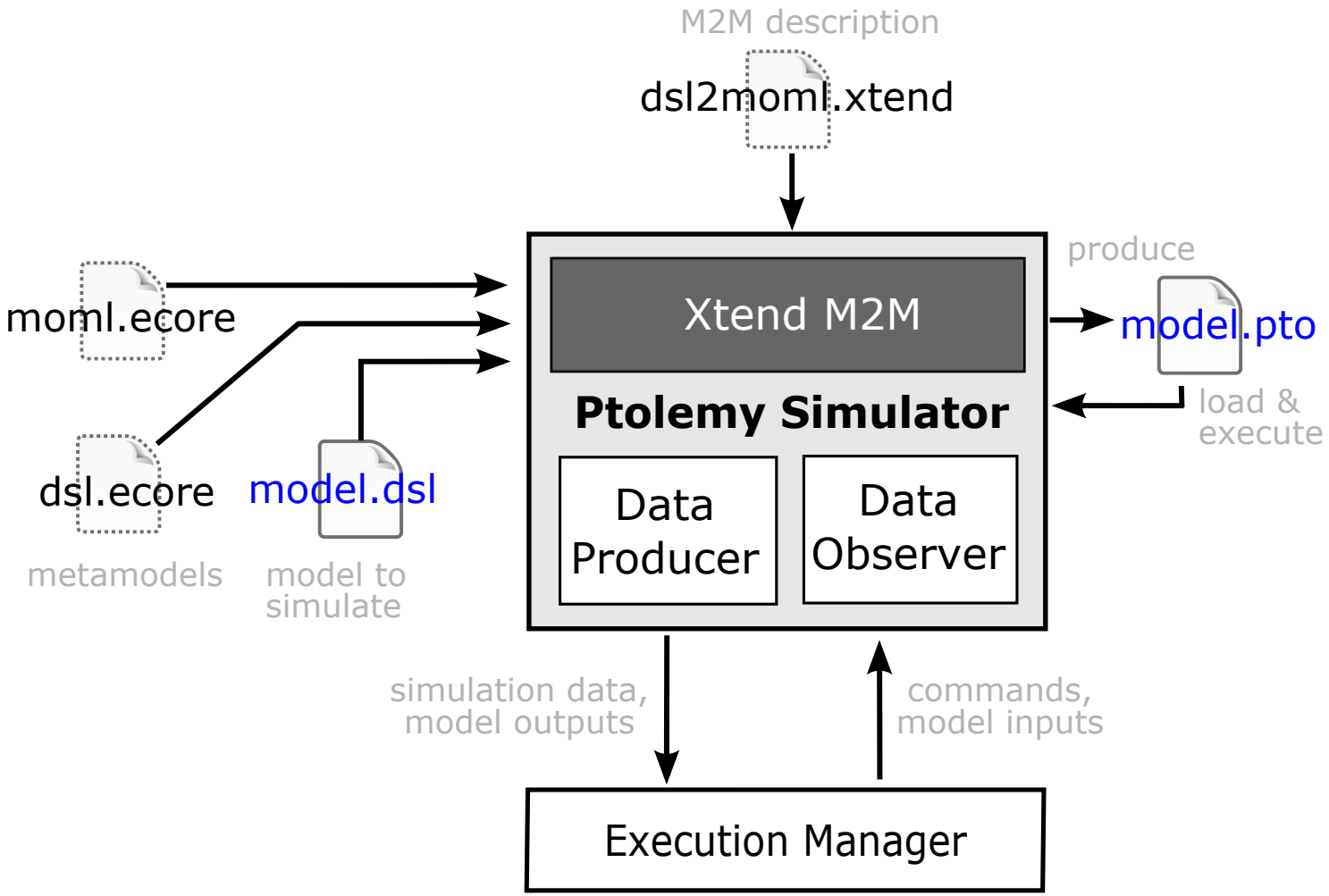
- Executable Models to describe behavior of reactive systems
- Ptolemy models are a set of interacting components $\rightarrow$ *Actor-Oriented Design*
- Constructed under a model of computation

# Ptolemy Actor Example

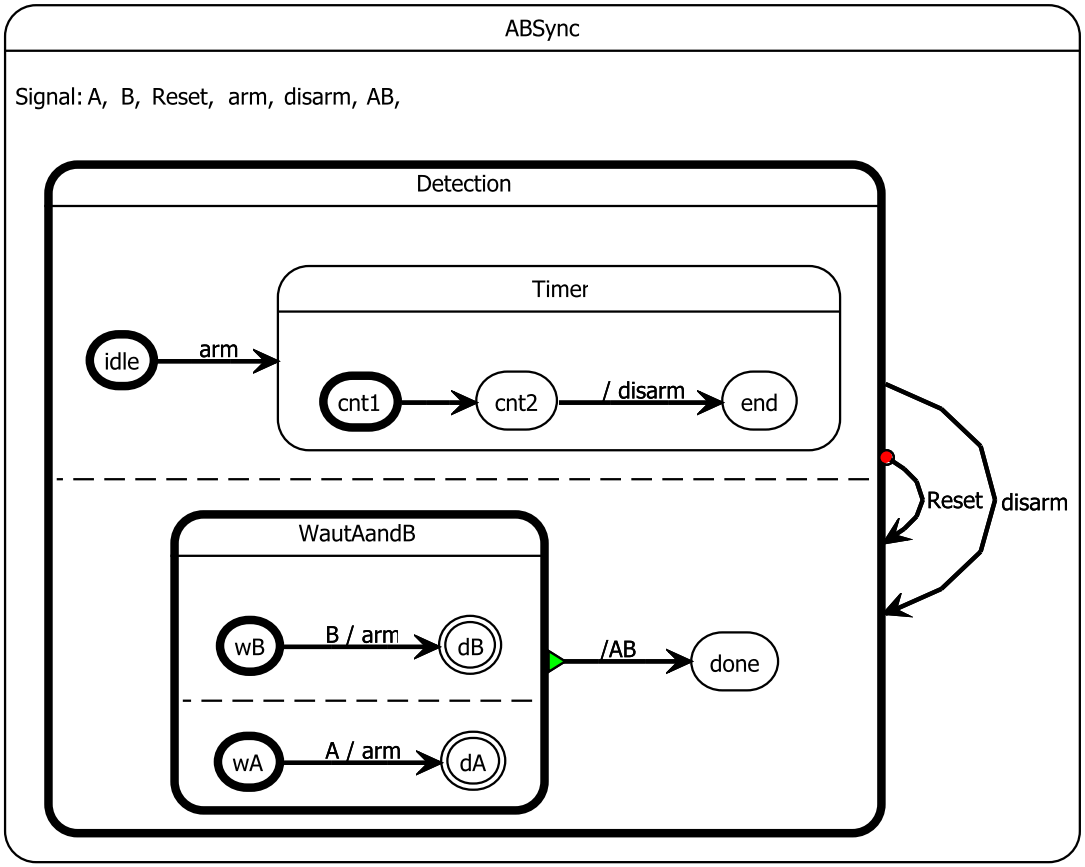



Introduction to Ptolemy II, UC Berkeley, 2008

# Schematic overview

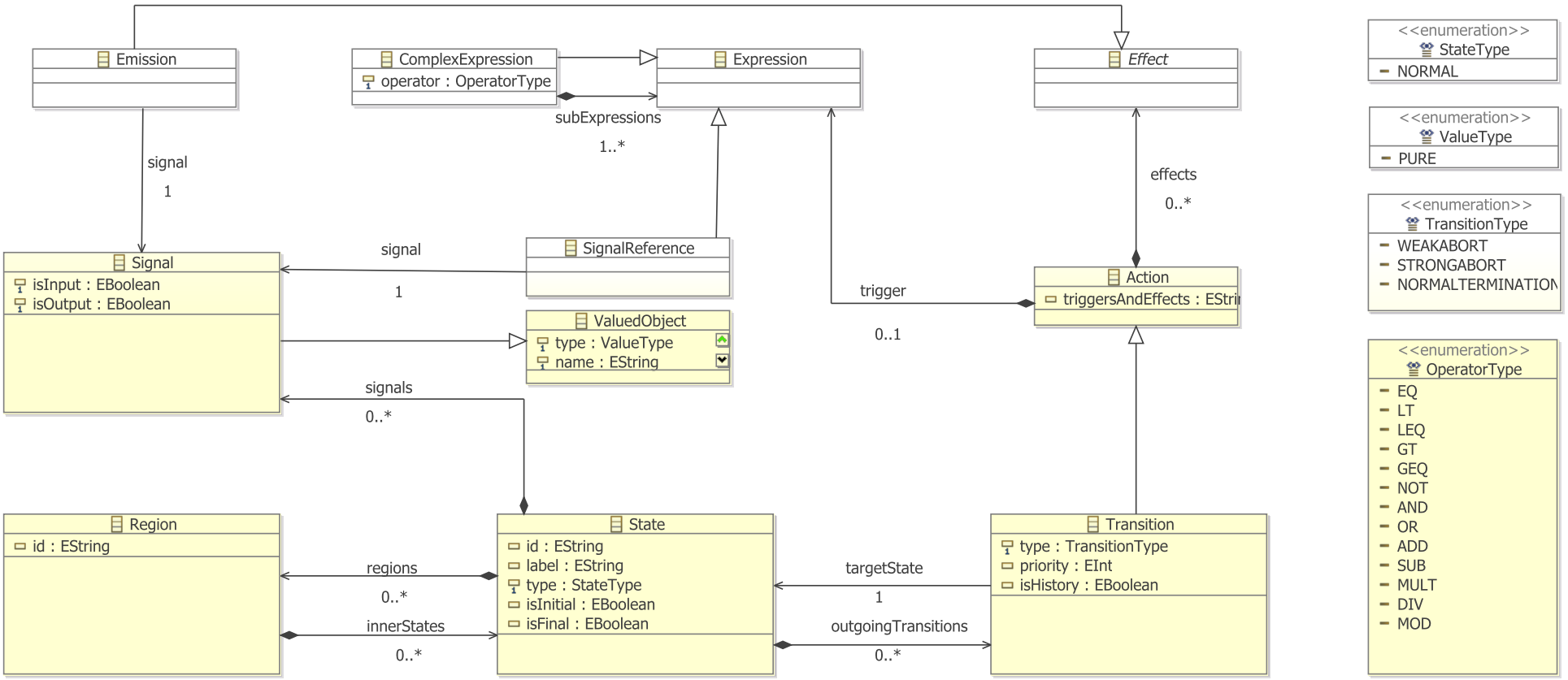


# SyncCharts




- Statechart dialect,
- Mealy machine with
  - Parallelism, hierarchy, compound events, broadcast
- Graphical notation for synchronous language Esterel
- Synchronous hypothesis
  - Discrete ticks
  - Computations take no time

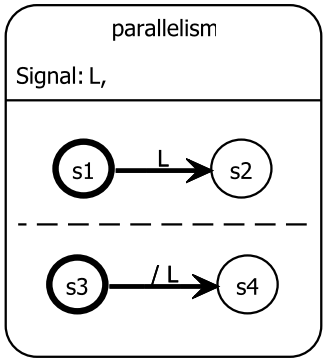
# Abstract Syntax (simplified)

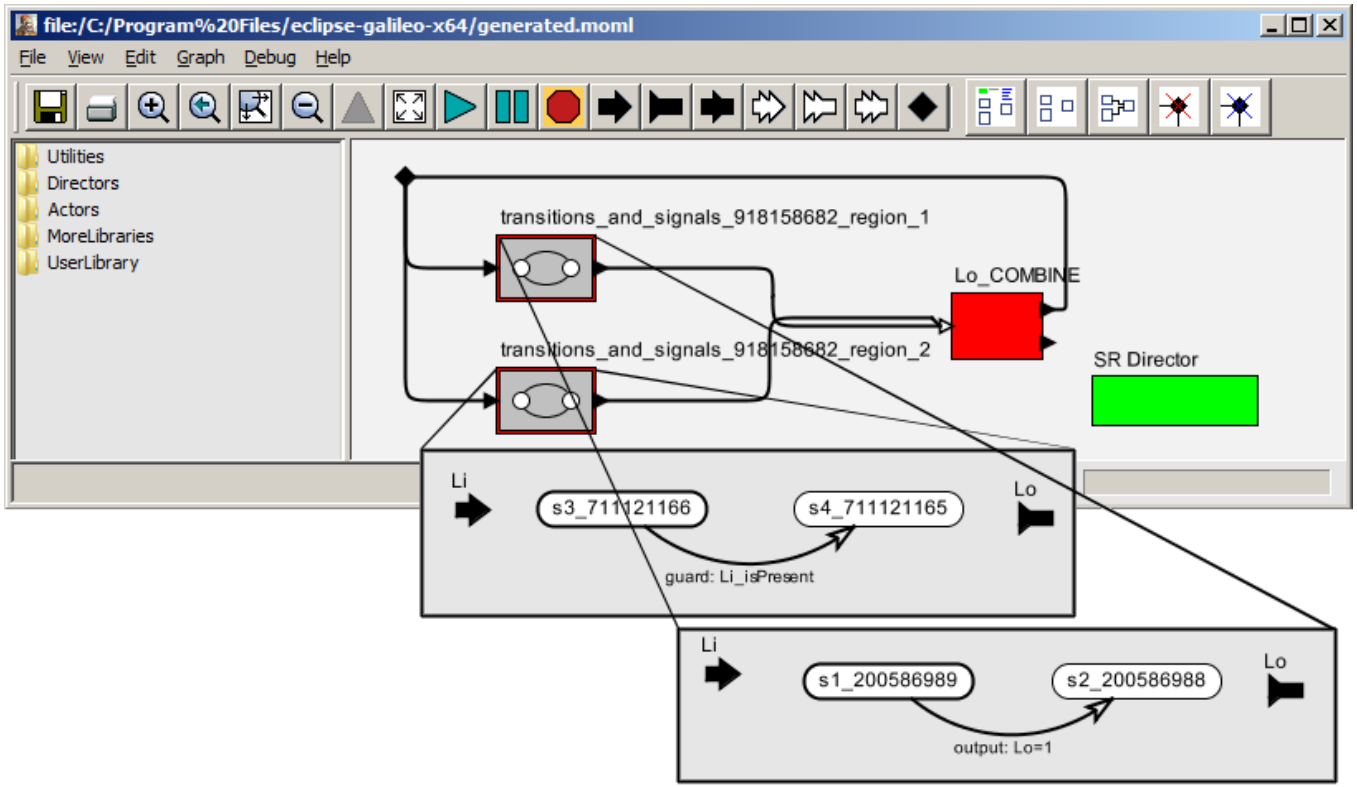


# Ptolemy Simulation Engine

- Ptolemy supports:
  - Modal model $\leftrightarrow$ mealy machine
  - Concurrency (inherently) $\leftrightarrow$ orthogonality
  - Compound actors $\leftrightarrow$ hierarchy
  - Expression language $\leftrightarrow$ compound events
- Interesting:
  - Explicit signal representation vs. broadcast
  - Signal assignment (must/cannot analysis)
  - Transition priorities
  - Normal termination

# Transformation Example: Parallelism and Signals

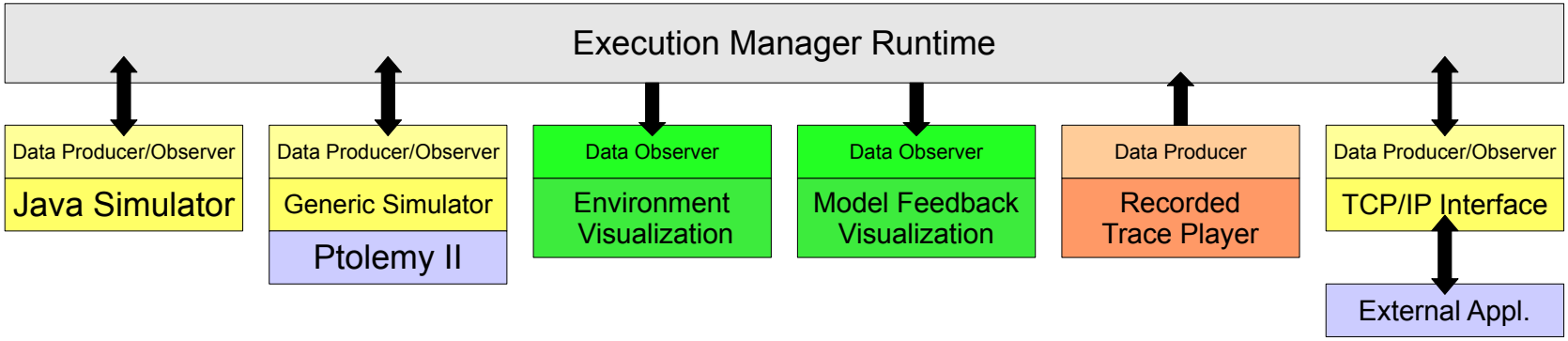


# Overview

- Model Semantics
  - Specification Options
  - KlePto Concept
  - Case Study: SyncCharts
- Model Execution
  - Framework Overview
  - User Interface
- Summary

# Architecture

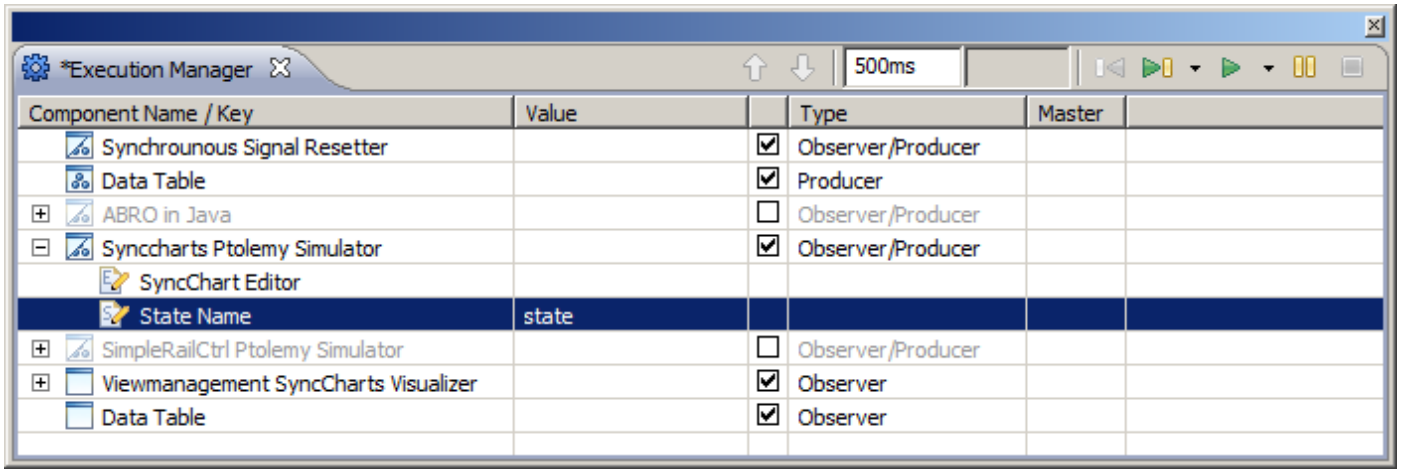



- DataComponent
- Producer vs. observer
- Interface: `step()` method
- *JSON* data exchange format

# Sample DataComponent

```
1  public class DataComponent extends JSONObjectDataComponent
2                              implements IJSONObjectDataComponent {
3     boolean doneI;
4
5     public void initialize() {
6        doneI = false;
7     }
8
9     public boolean isObserver() {return true;}
10    public boolean isProducer() {return true;}
11
12    public JSONObject step(JSONObject jSONObject)
13                   throws KiemExecutionException {
14       JSONObject returnObj = new JSONObject();
15       if (!doneI && jSONObject.has("I")
16          && (JSONSignalValues.isPresent(jSONObject.get("I")))) {
17          //change state to doneI when signal I is present
18          doneI = true;
19          //output signal O
20          returnObj.accumulate("O", JSONSignalValues.newValue(true));
21       }
22       return returnObj;
23    }
24 }
```

# KIEM Eclipse View




- Linear scheduling
- JSON data pool
- Execution control
- Add, remove, disable, properties, save, load, ...

# KIELER KlePto Simulation Demo

**LIVE DEMO**

# Summary

- KlePto
  1. Ptolemy integration in Eclipse
  2. Construct runnable Ptolemy models for EMF based models
  3. Use Xtend M2M transformations
- Execution Manager
  - Infrastructure for interactive model execution
  - Simple interface using standard data representation
  - Intuitive UI
  - Also: Visualization, stepwise transformation, model checking, online debugging, regression tests ...

# To Go Further


- MOTIKA, C.
  Semantics and execution of domain specific models—KlePto and an execution framework.
  http://rtsys.informatik.uni-kiel.de/~biblio/downloads/theses/cmot-dt.pdf.
- UC BERKELEY, EECS DEPT.
  Ptolemy webpage.
  http://ptolemy.eecs.berkeley.edu/.
- UNI KIEL, REAL-TIME AND EMBEDDED SYSTEMS GROUP.
  KIELER webpage.
  http://www.informatik.uni-kiel.de/en/rtsys/kieler/.