ESWG #1

UC Berkeley
Mobies Technology Project

Process-Based Software Components for Networked Embedded Systems

PI: Edward Lee

CoPI: Tom Henzinger

Goals

- **Project Goals**
  - This project focuses on software creation for embedded systems with emphasis on systematically handling cross-cutting concerns such as safety, real-time, concurrency, liveness, adaptability, and integratability. The approach is component based, with compositional semantics and system-level type systems.
- **Research Area**
  - Heterogeneous modeling and design, code generation
- **Targeted Development Context**
  - The Ptolemy II tool will be used for modeling, simulation, and design of concurrent, real-time, embedded systems.

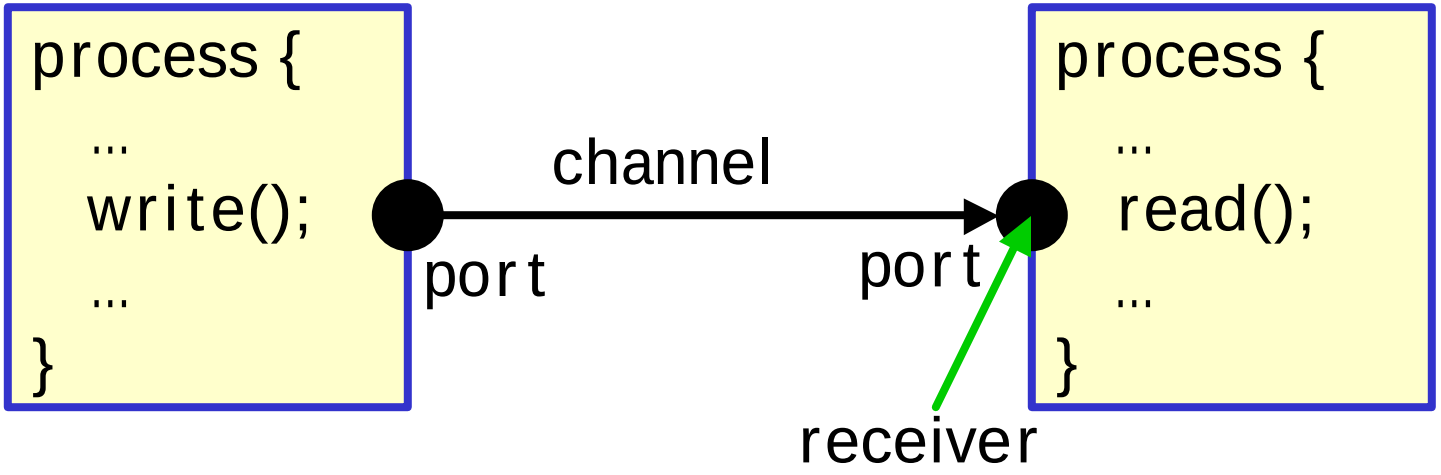
Products

- Ptolemy II
  - Java-based framework
  - Open source
  - Open architecture
  - Platform for domain-specific languages
  - Toolkit for code generators
  - GUI toolkit

  } Preliminary at this time
- Domains
  - Continuous-time
  - Discrete-time
  - Discrete events
  - Communicating sequential processes
  - Process networks
  - Dataflow
  - Time-triggered
  - 3-D graphics
  - Synchronous/reactive
  - Publish/Subscribe

  } Preliminary at this time

Producer / Consumer



- Are actors active? passive? reactive?
- Are communications timed? synchronized? buffered?

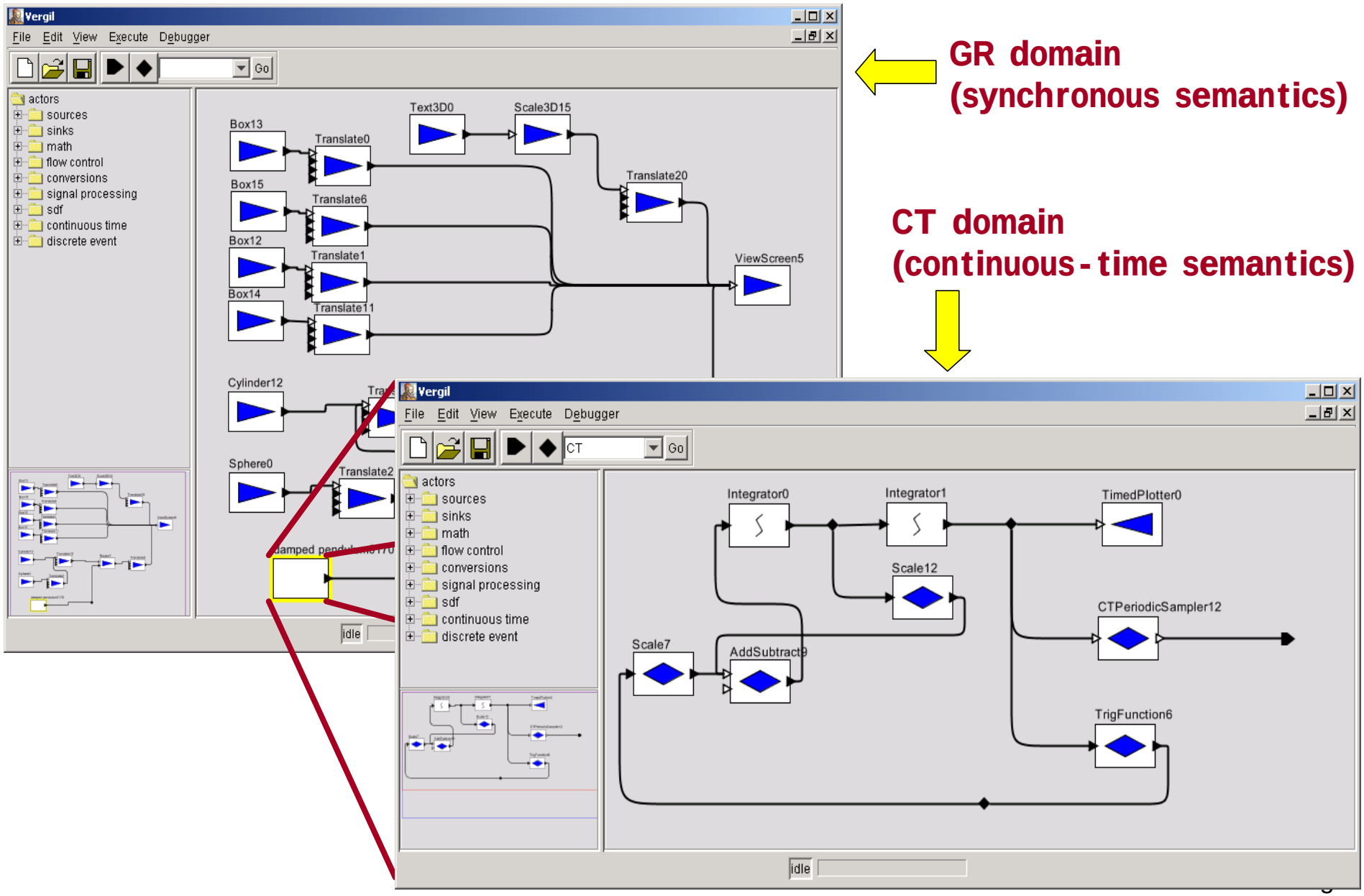
A *domain* answers these questions in a disciplined way

Interfaces

- **Design Data**
  - MoML XML schema
  - Configurable GUI toolkit
  - Java API
  - CORBA components } Preliminary
- **Components for Component-Based Design**
  - Domain polymorphic
  - Data polymorphic
  - Type constraint propagation
  - Structured types (records, arrays)
  - System-level types
  - Real-time types

  } Preliminary

Ptolemy II Visual Syntax



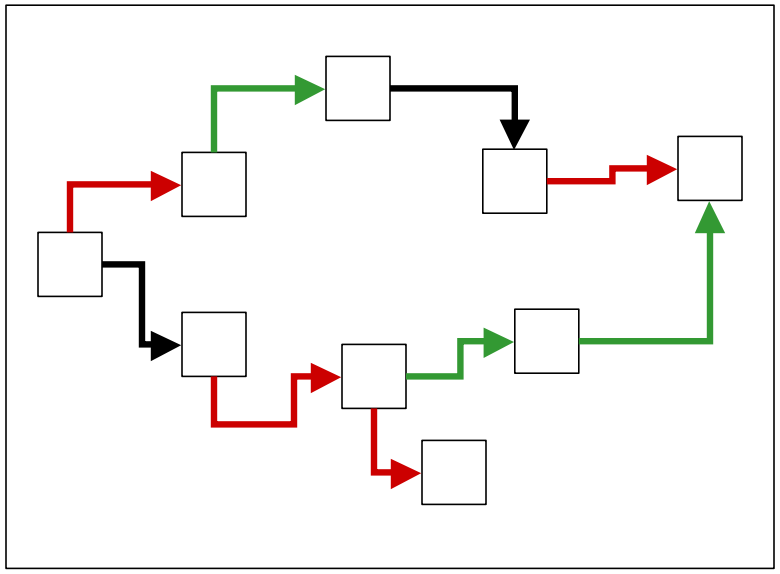
GR domain
(synchronous semantics)

CT domain
(continuous-time semantics)

Hierarchical Heterogeneity

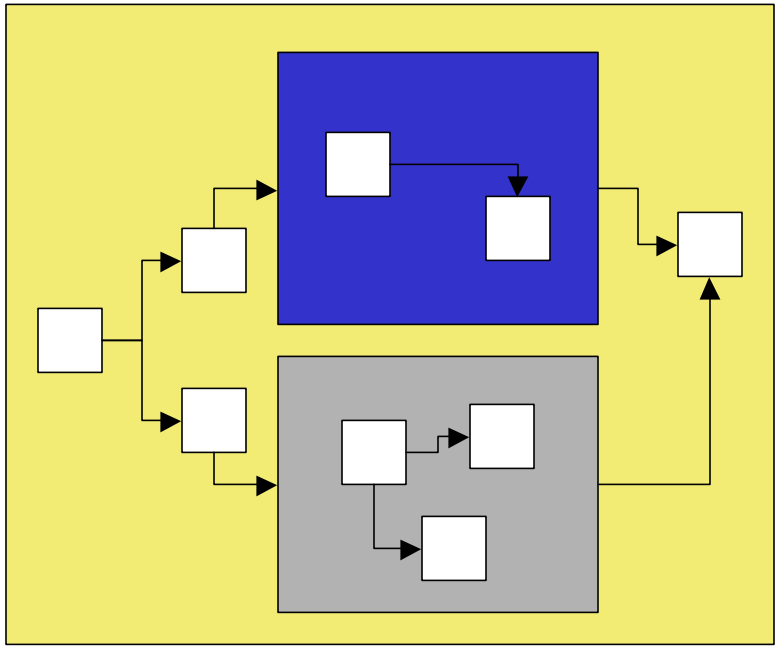
vs. Amorphous Heterogeneity

Amorphous



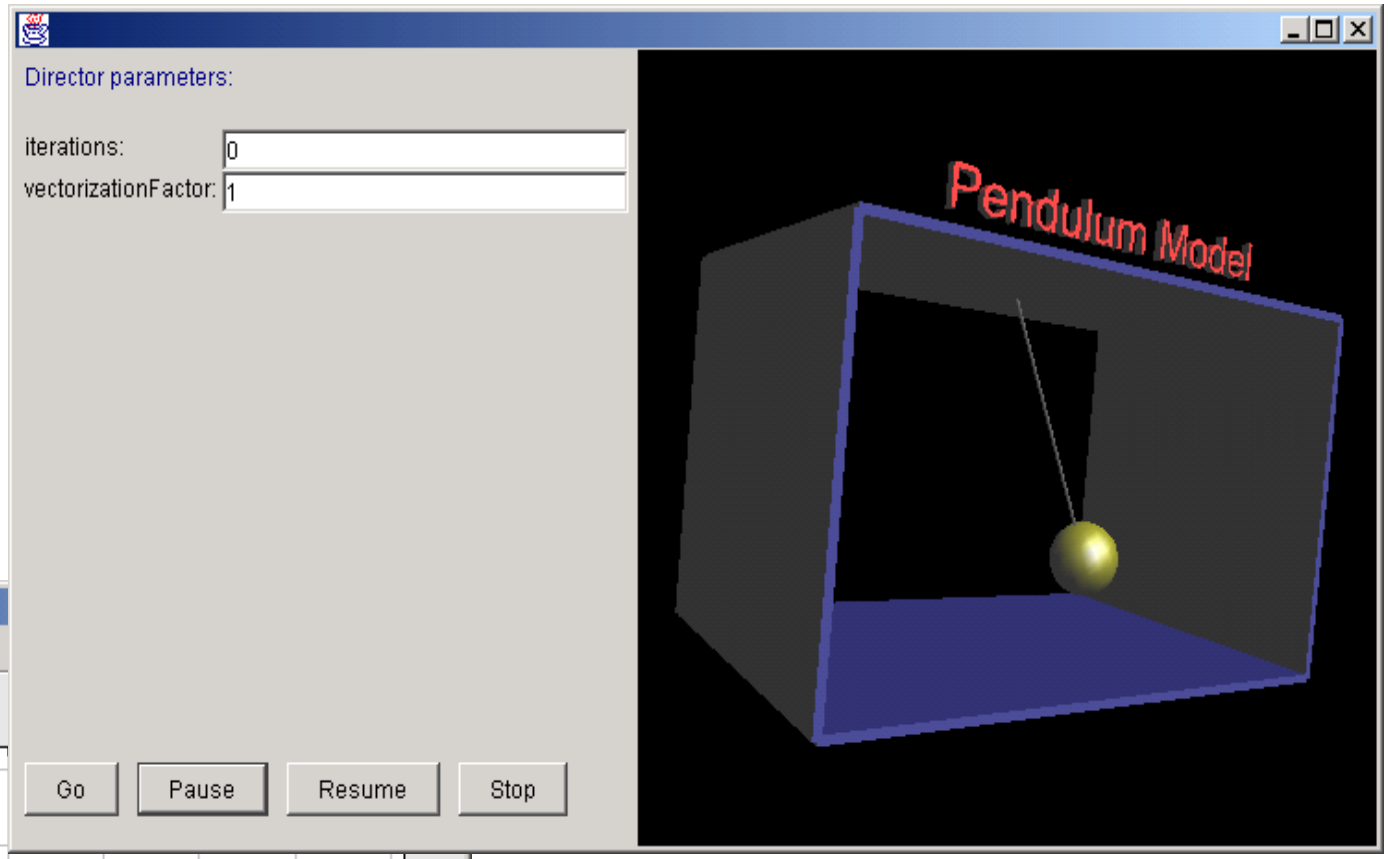
Color is a communication protocol only, which interacts in unpredictable ways with the flow of control.

Hierarchical

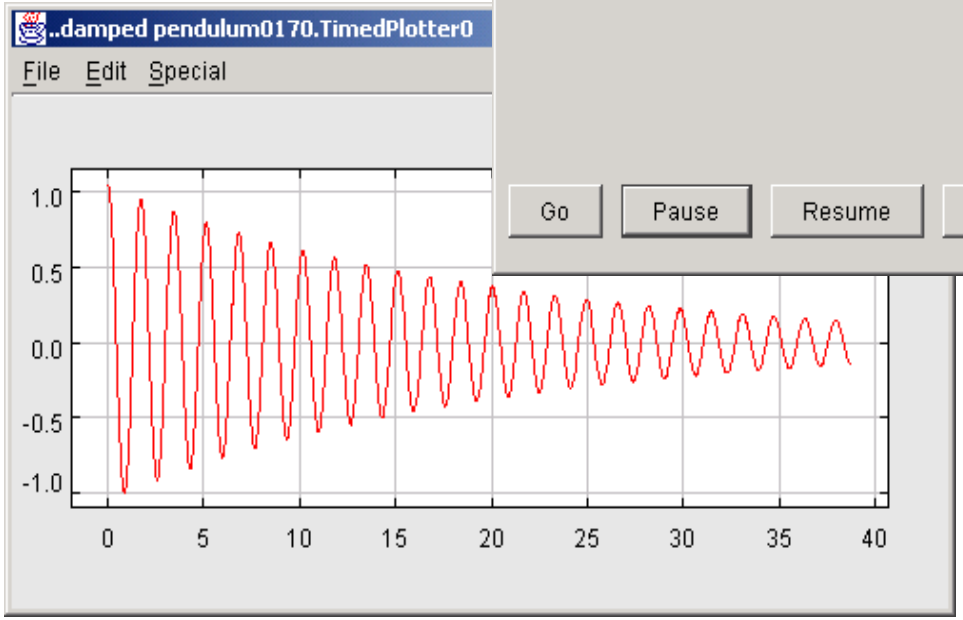


Color is a domain, which defines both the flow of control and interaction protocols.

Example Execution



3-D Graphics



PtPlot tool

Lee & Henzinger

Example Domain

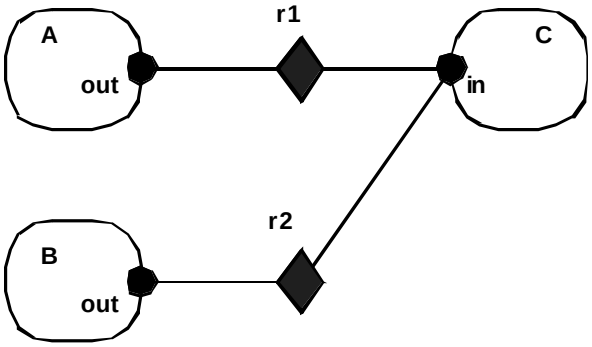
- **Giotto domain**
  - Synchronous semantics
  - Concurrent finite-state machines
  - Time triggered, multirate
  - Verifiable semantics

- **Realization in Ptolemy II**
  - About 1000 lines of code/comments in four classes
    - GiottoDirector
    - GiottoScheduler
    - GiottoReceiver
    - GiottoActorComparator
  - Leverages pre-existing FSM domain.
  - Leverages pre-existing GUI and actor library.

MoML Specification

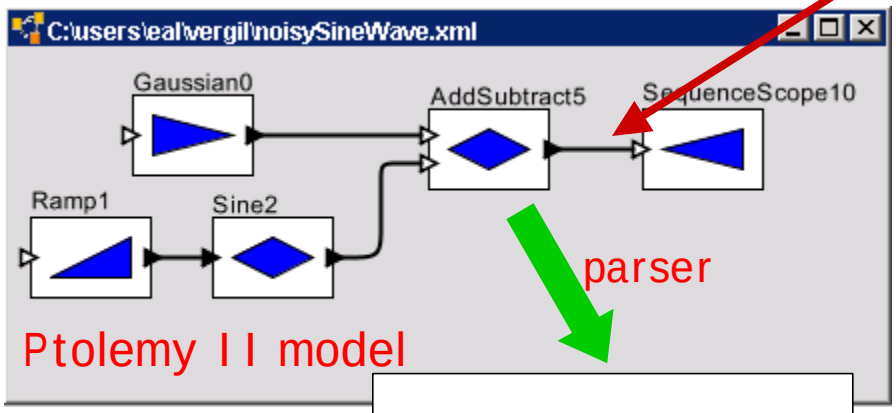
MoML is an XML schema representing *only* the abstract syntax of component-based designs.

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE model SYSTEM "DTD location">
<model class="classname">
  <entity name="A" class="classname"></entity>
  <entity name="B" class="classname"></entity>
  <entity name="C" class="classname"></entity>
  <relation name="r1"></relation>
  <relation name="r2"></relation>
  <link port="A.out" relation="r1"/>
  <link port="B.in" relation="r1"/>
  <link port="C.out" relation="r2"/>
  <link port="B.in" relation="r2"/>
</model>
```



Code Generators

Domain semantics defines communication, flow of control



Ptolemy II model

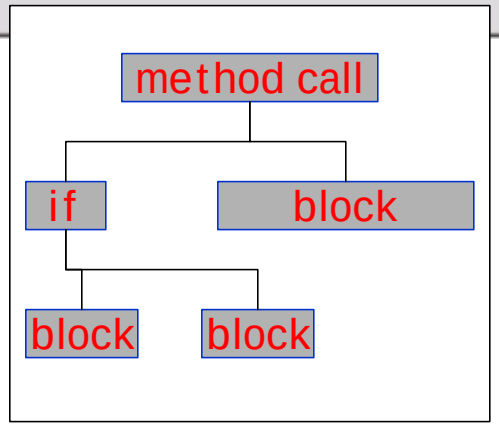
parser

scheduler

Schedule:
- fire Gaussian0
- fire Ramp1
- fire Sine2
- fire AddSubtract5
- fire SequenceScope10

code generator

All actors will be given in Java, then translated to embedded Java, C, VHDL, etc.



abstract syntax tree

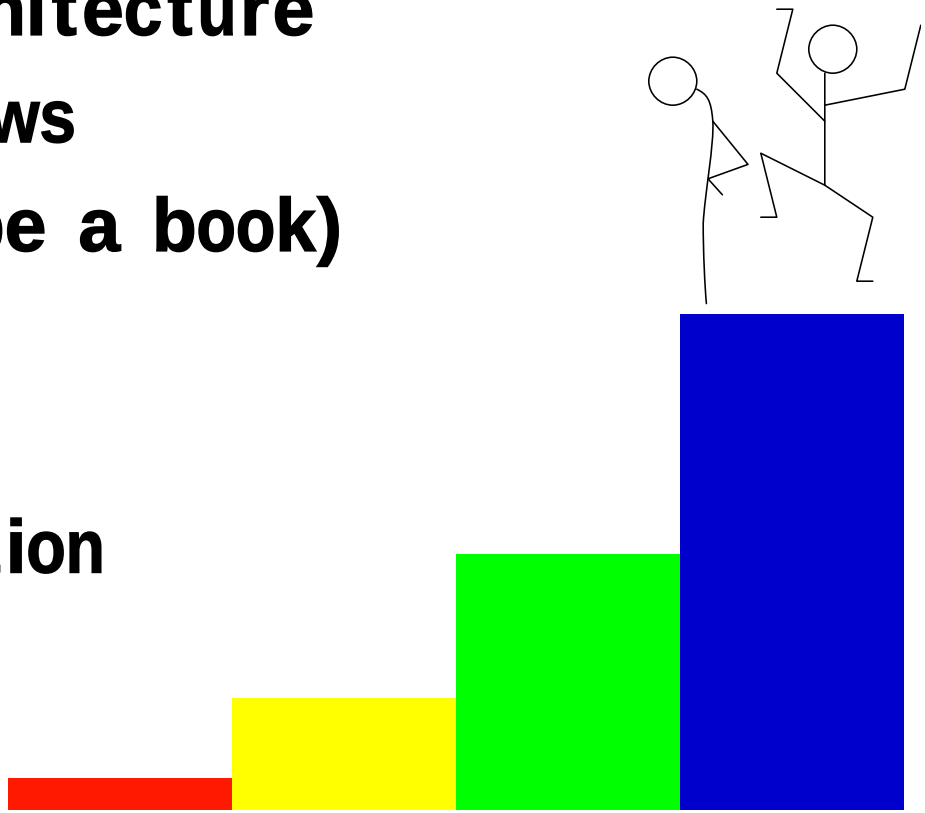
```
…
for (int i = 0; i < plus.getWidth(); i++) {
  if (plus.hasToken(i)) {
    if (sum == null) {
      sum = plus.get(i);
    } else {
      sum = sum.add(plus.get(i));
    }
  }
}
…
```

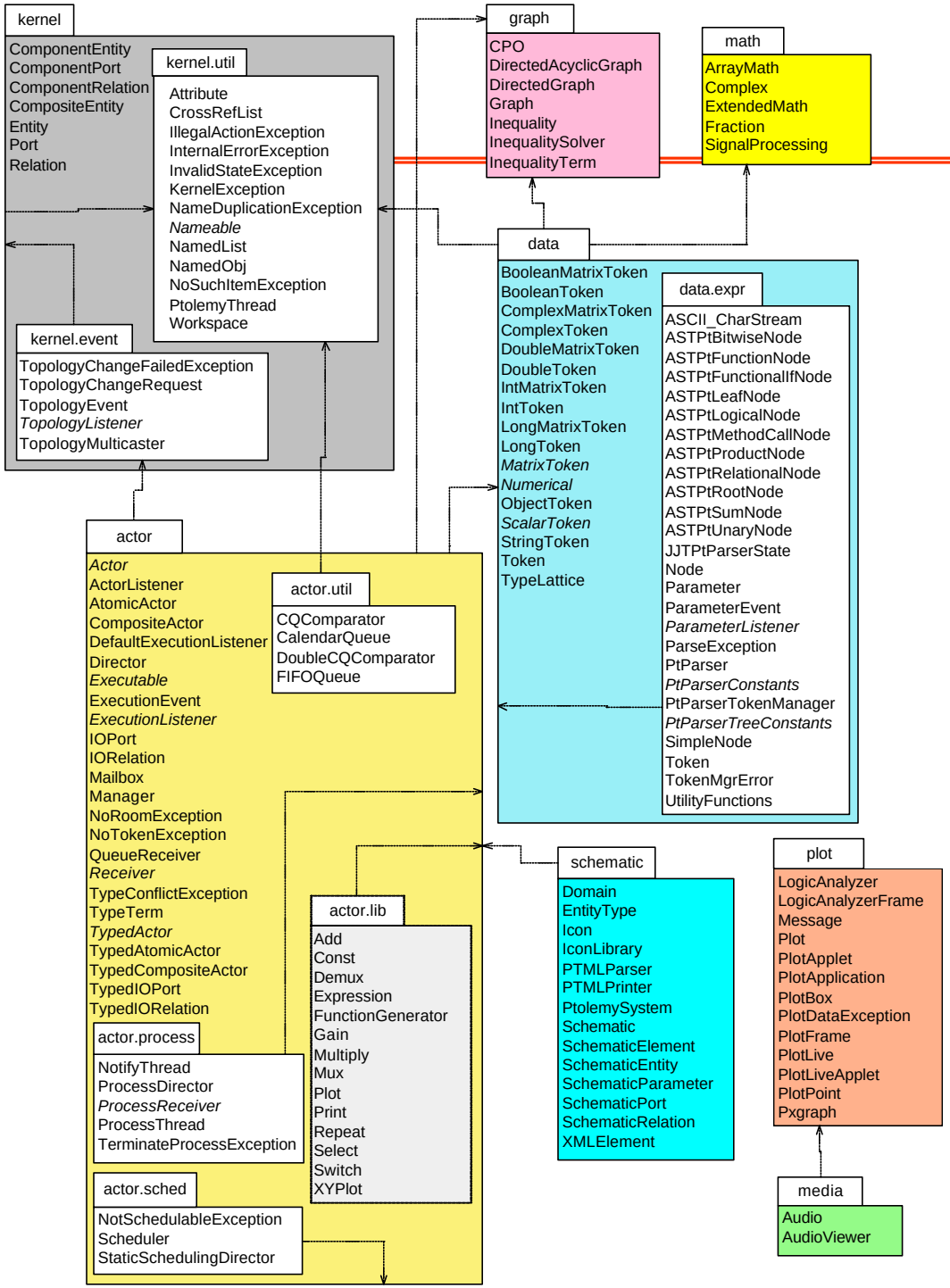
target code

First version created by Jeff Tsay.

Software Practice

- Object models in UML
- Design patterns
- Careful package organization
- Layered software architecture
- Design and code reviews
- Design document (to be a book)
- Nightly build
- Regression tests
- Sandbox experimentation
- Code rating





Ptolemy II Packages

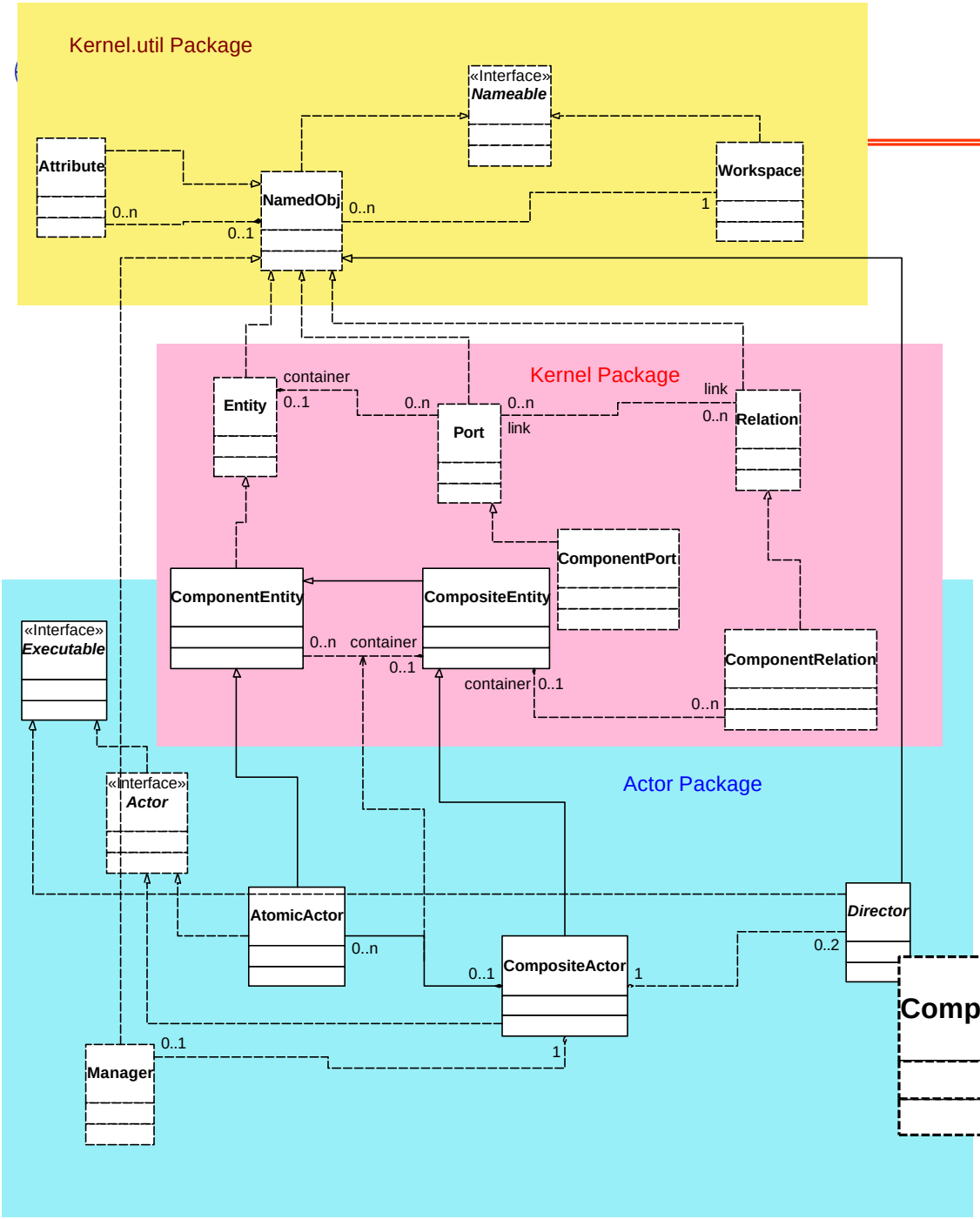
- kernel (clustered graphs)
- actor (executable models)
- data (tokens, expressions)
- vergil (API for UIs)
- graph (graph algorithms)
- math (math algorithms)
- plot (plotting utilities)
- domains (modeling framelets)

kernel

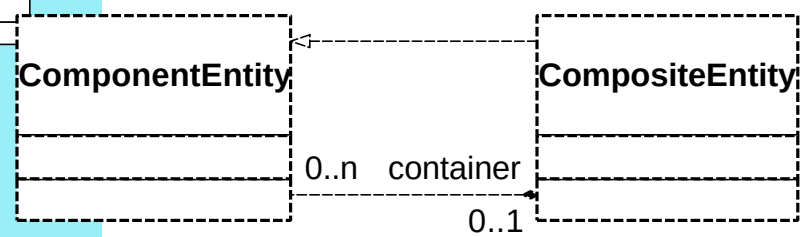
ComponentEntity
ComponentPort
ComponentRelation
CompositeEntity
Entity
Port
Relation

Lee & Henzinger

Ptolemy II Key Classes



UML static structure diagram for the key classes in the kernel, kernel.util, and actor packages.



Lee & Henzinger

Code Rating & Coverage

Ptolemy II Internal Website - Netscape

File Edit View Go Communicator Help

Back Forward Reload Home Search Netscape Print Security Shop Stop

Bookmarks Location: http://ptolemy.eecs.berkeley.edu/~ptII/ What's Related

Instant Message WebMail Contact People Yellow Pages Download Channels Welcome to Agil

Ptolemy Group Internal Page

Contents

Nightly home
Internal home

Mail
Log
Script
Code Coverage

Package .ptolemy.kernel

Compilation Units	method	constructor	basic block	branch	switch	catc
ComponentEntity.java	8/8=100%	3/3=100%	23/26=88%	19/22=86%	0/0=N/A	0/0=N
ComponentPort.java	30/30=100%	3/3=100%	101/116=87%	74/92=80%	0/0=N/A	0/4=(
ComponentRelation.java	8/8=100%	3/3=100%	28/34=82%	26/32=81%	0/0=N/A	0/0=N
CompositeEntity.java	29/29=100%	3/3=100%	119/136=87%	93/104=89%	0/0=N/A	0/8=(
Entity.java	17/17=100%	4/4=100%	53/56=94%	28/32=87%	0/0=N/A	0/2=(
Port.java	17/17=100%	3/3=100%	65/73=89%	47/56=83%	0/0=N/A	0/4=(
Relation.java	10/10=100%	4/4=100%	35/40=87%	19/20=95%	0/0=N/A	0/5=(
TOTAL	119/119=100%	23/23=100%	424/481=88%	306/358=85%	0/0=N/A	0/23=

Package .ptolemy.kernel.util

Compilation Units	method	constructor	basic block	branch	switch
Attribute.java	4/4=100%	3/3=100%	18/20=90%	16/18=88%	0/0=N/A
ChangeListener.java	0/0=N/A	0/0=N/A	0/0=N/A	0/0=N/A	0/0=N/A
ChangeRequest.java	3/5=60%	1/1=100%	9/14=64%	5/14=35%	0/0=N/A
CrossRefList.java	17/17=100%	6/6=100%	66/75=88%	66/78=84%	0/0=N/A
DebugEvent.java	0/0=N/A	0/0=N/A	0/0=N/A	0/0=N/A	0/0=N/A

Ptolemy Group, Department of EECS, UC Berkeley - Comments to: eallocal@ptolemy.eecs.berkeley.edu, Copyright © 1999-2000.

Document: Done

Nightly build creates a web page displaying code rating, code coverage of regression tests, and results of the build and the tests.

Lee & Henzinger

Interaction Mechanisms

- Software releases
  - December 2000 (0.5 beta)
  - February 2001 (0.5)…
- CVS access over the internet
  - For the brave, with whom we work closely
- Kluwer book (est. March 2001)
  - Developers guide
  - UML object models of everything
- Ptolemy Miniconference
  - March 22-23, 2001 Claremont Hotel, Oakland

Near Term Schedule

- **Software releases**
  - December 2000 (0.5 beta)
  - February 2001 (0.5)
  - December 2001 (0.6 beta)
  - February 2002 (0.6)

- **Milestones**
  - Record types (February 2001)
  - Miniconference (March 2001)
  - System-level type definitions (Summer 2001)
  - Real-time type definitions (Summer 2002)
  - …

Releasability Restrictions

- None
  - Source code provided.
  - Can be commercialized.
  - Can be used in commercial products.

Required from OEP

Domain definitions

- What is a component?
- How do components interact?
- Disciplined interactions, for
  - understandability
  - verifiability