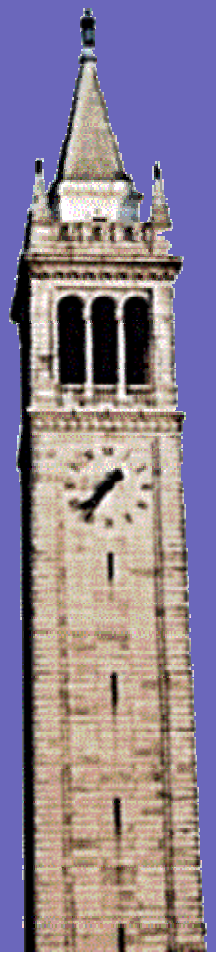
Process-Based Software Components for Networked Embedded Systems



Edward A. Lee, PI

UC Berkeley

Core Technical Team (Mobies, SEC, and GSRC):

Christopher Hylands, Mary P. Stewart, Joern Janneck, Sonia Sachs, Elaine Cheong, Chamberlain Fong, Jie Liu, Xiaojun Liu, Stephen Neuendorffer, Brian K. Vogel, Paul Whitaker, Yuhong Xiong

Mobies PI Meeting

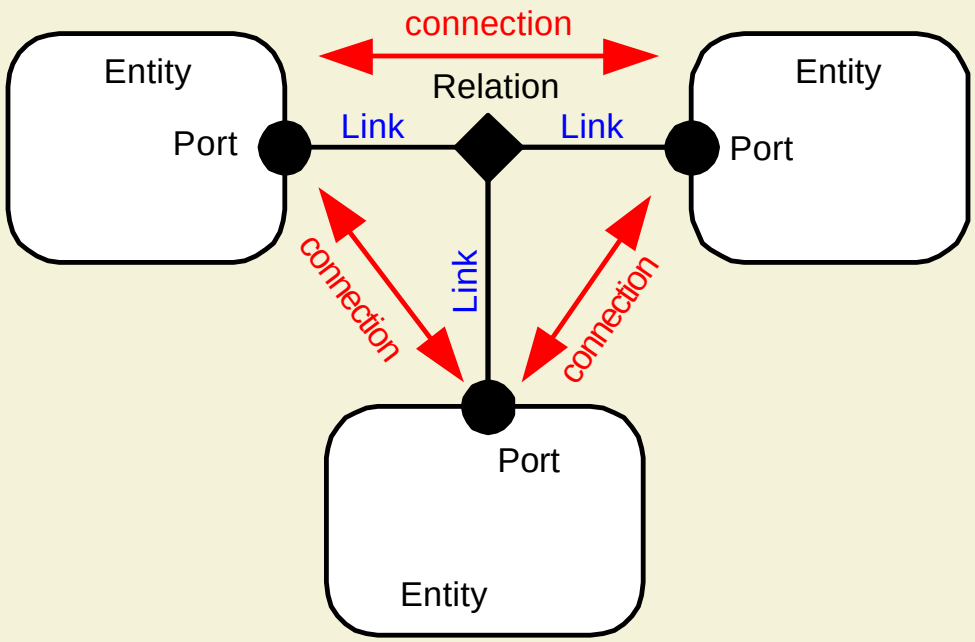
New Orleans, January 23-25, 2001

Status Update

- **Ptolemy II version 1.0 alpha available soon**
  - A platform that can promote collaboration
  - Open source, open architecture
  - Rated code (red, yellow, green)
  - Core code is very high quality (green)
  - Code is written to be read
  - Extensible GUI (all red, currently)

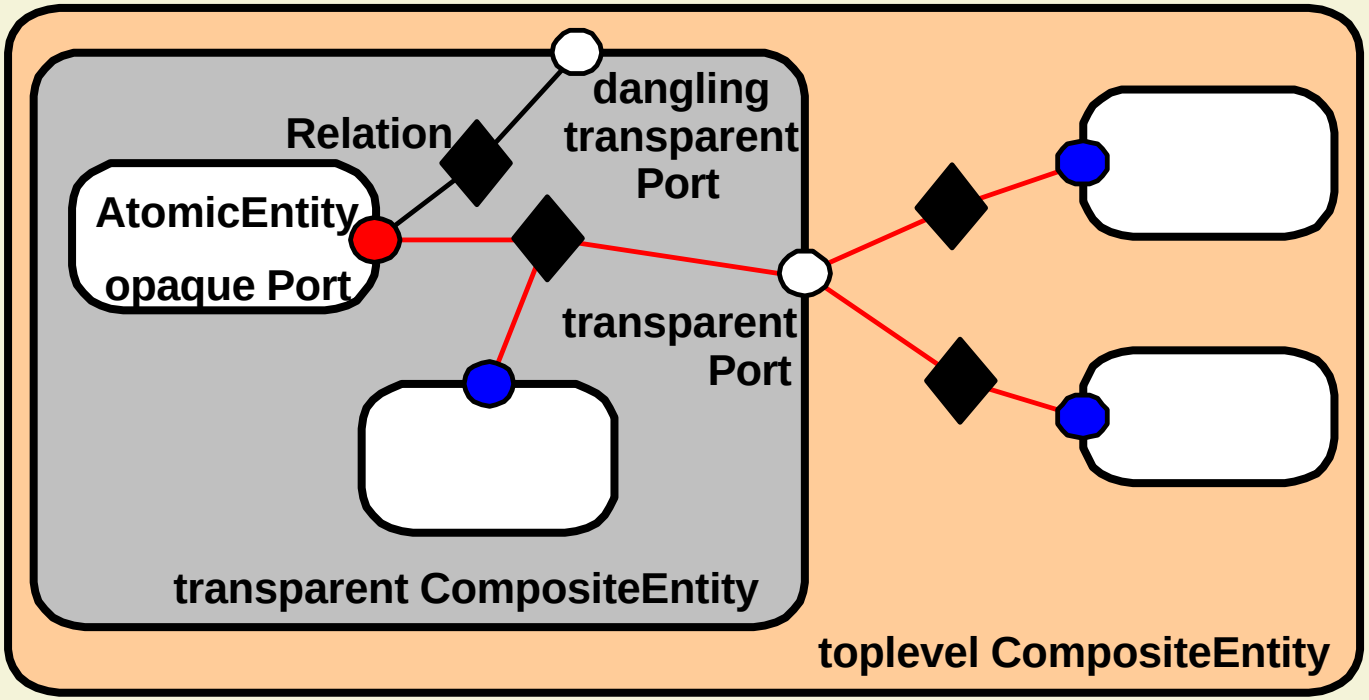
- **Commercial support organization**
  - recently formed: Agile Design

Components and their Relationships



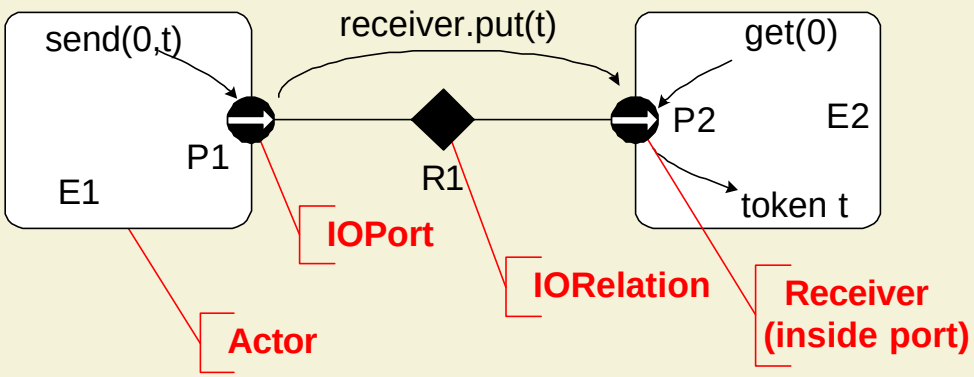
The Ptolemy II kernel provides an *abstract syntax* - clustered graphs - that is well suited to a wide variety of component-based modeling strategies, ranging from state machines to process networks.

Hierarchy - Construct components from finer grain components.



Actor Package – Infrastructure for Producer/Consumer Components

Basic Transport:



Services in the Infrastructure:

- broadcast
- multicast
- busses
- mutations
- clustering
- parameterization
- typing
- polymorphism

Domains – Provide semantic models for component interactions

- CSP – concurrent threads with rendezvous
- CT – continuous-time modeling
- DE – discrete-event systems
- DDE – distributed discrete events
- FSM – finite state machines
- DT – discrete time (cycle driven) *
- Giotto – synchronous periodic *
- PN – process networks
- RTOS – priority-driven reactive models *
- SDF – synchronous dataflow
- SR – synchronous/reactive **

* New domains in Ptolemy II

** Not yet available in Ptolemy II

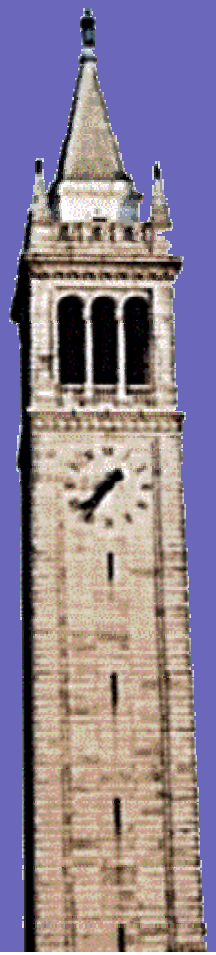
Ptolemy II – Our Software Laboratory

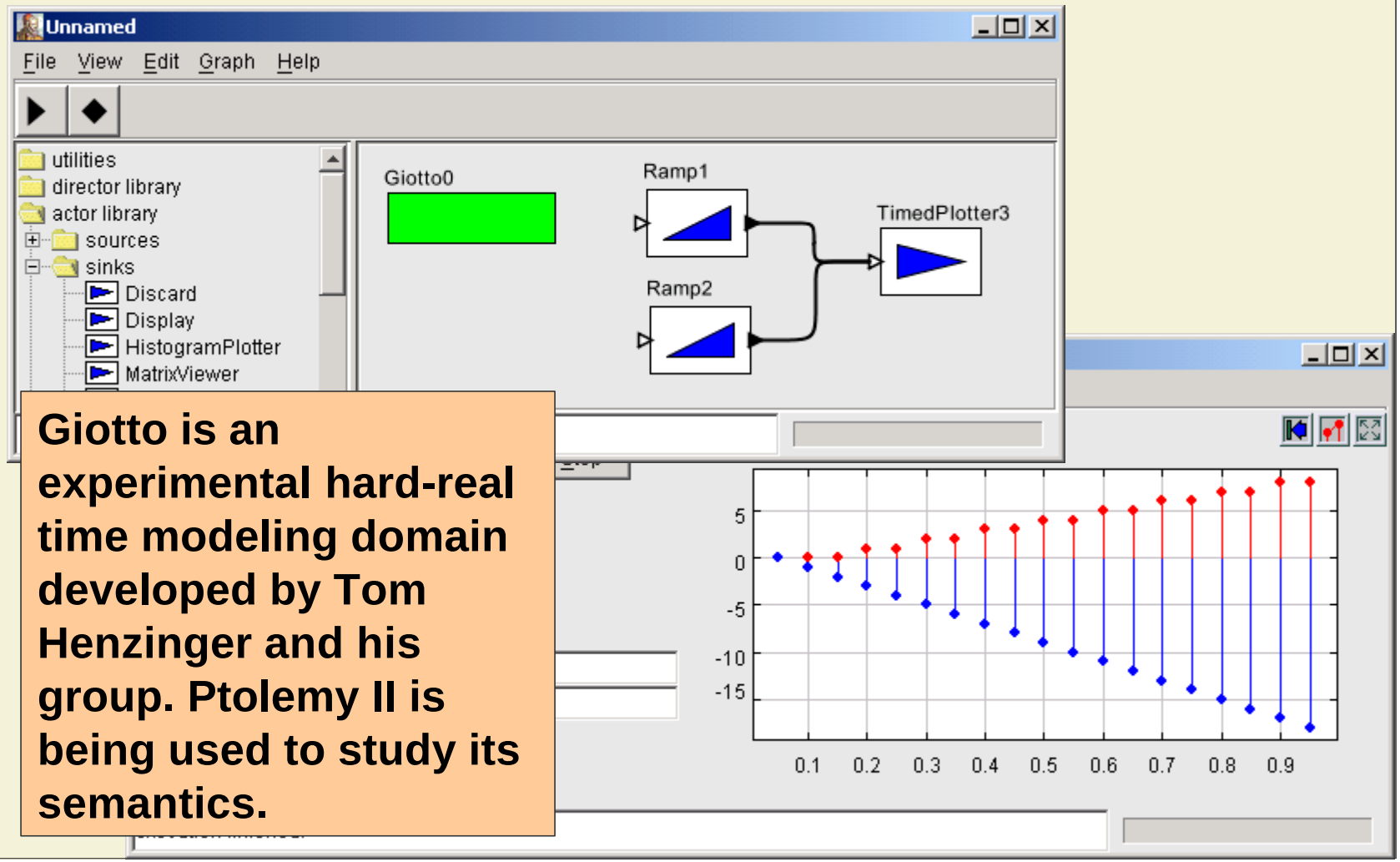
Ptolemy II –

Emphasis is on building a framework supporting experimentation with models of computation and their interactions.

http://ptolemy.eecs.berkeley.edu

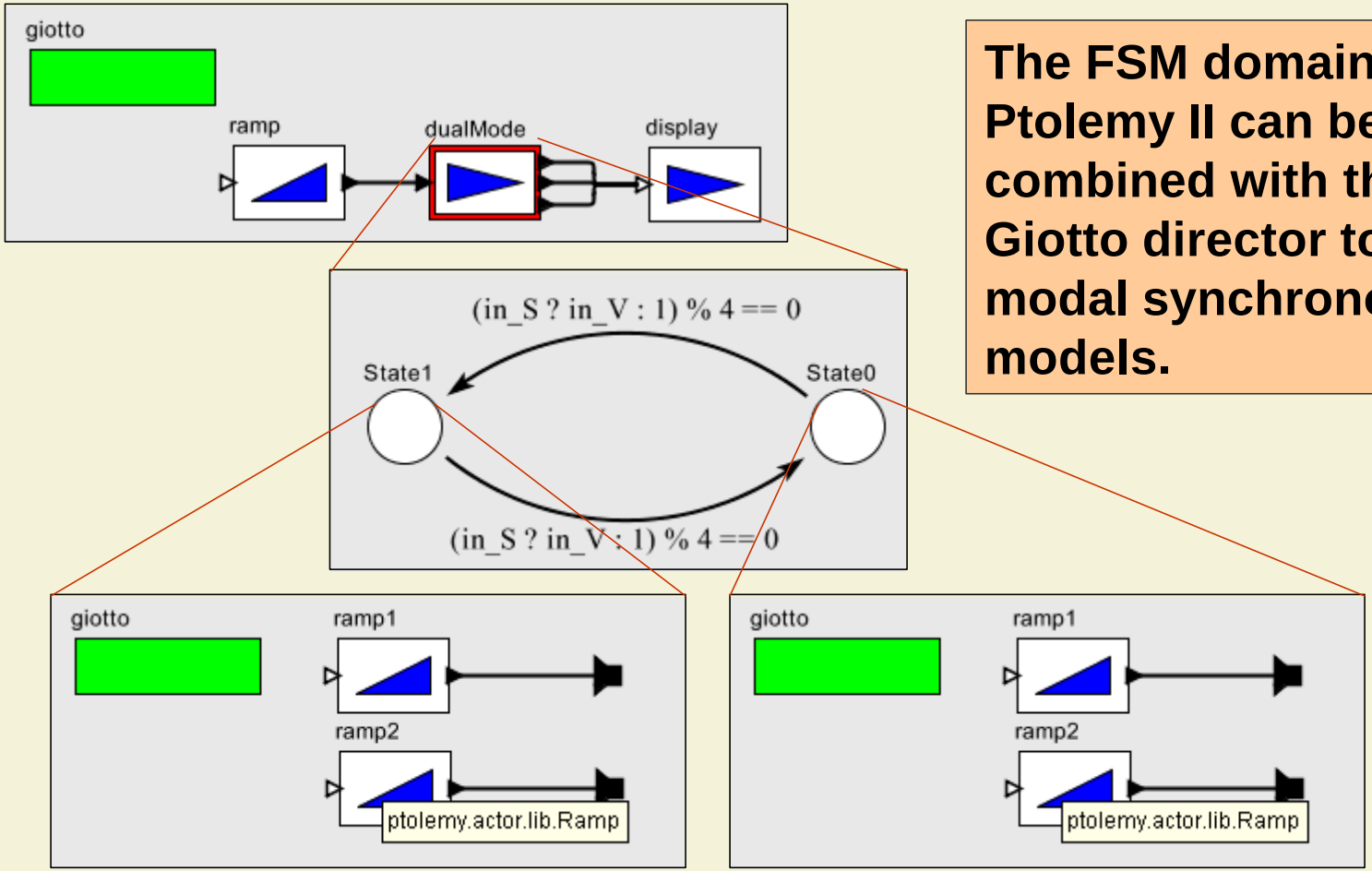
Exploring Giotto Semantics





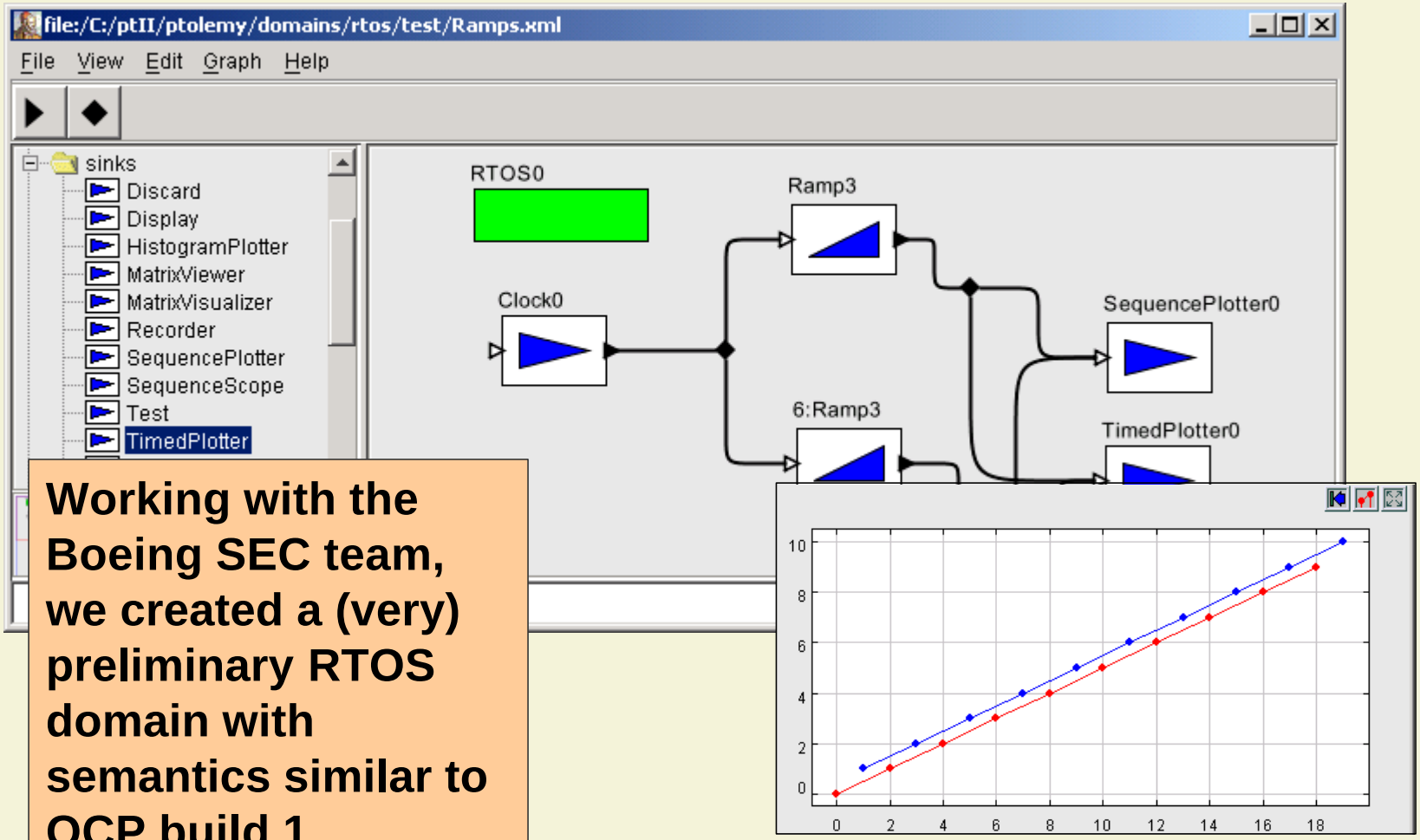
Giotto is an experimental hard-real time modeling domain developed by Tom Henzinger and his group. Ptolemy II is being used to study its semantics.

Modal Giotto Model



The FSM domain in Ptolemy II can be combined with the Giotto director to get modal synchronous models.

Exploring Hierarchical RTOS Semantics



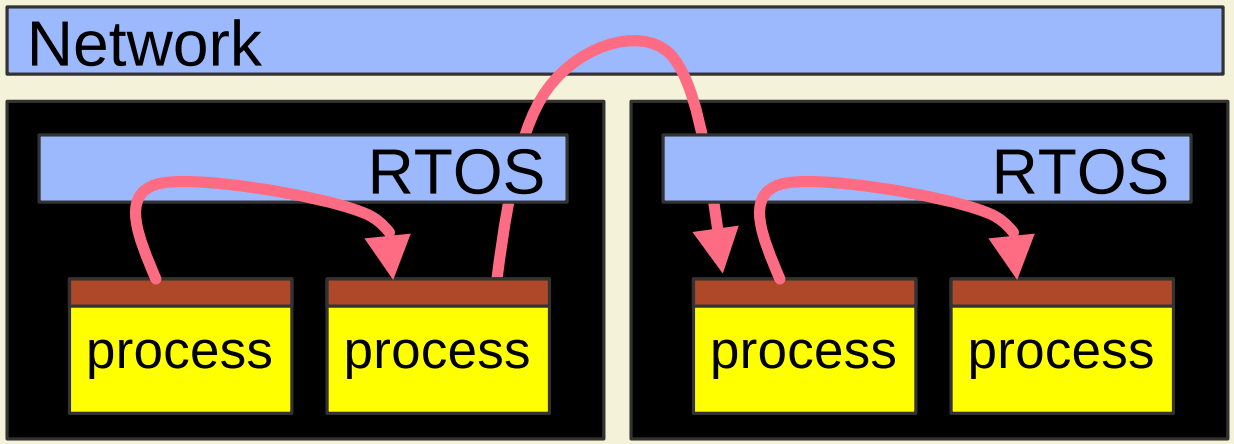
Working with the Boeing SEC team, we created a (very) preliminary RTOS domain with semantics similar to OCP build 1.

Hierarchical Real-Time Scheduling

- **Non preemptive**
  - + Allows precise interactions with other domains
  - + Allows precise mode changes
  - + (More) predictable behavior
  - - Requires fine-grain partitioning of components

- **Issues**
  - Document precise semantics
    - How are equal priorities handled?
  - Plug-in API for scheduling
    - Dynamic priorities & soft real time
  - Delegation of coarse-grain computations
    - To classical real-time processes (?)

RTOS-Based Design is a Fixed Three-Level Hierarchy

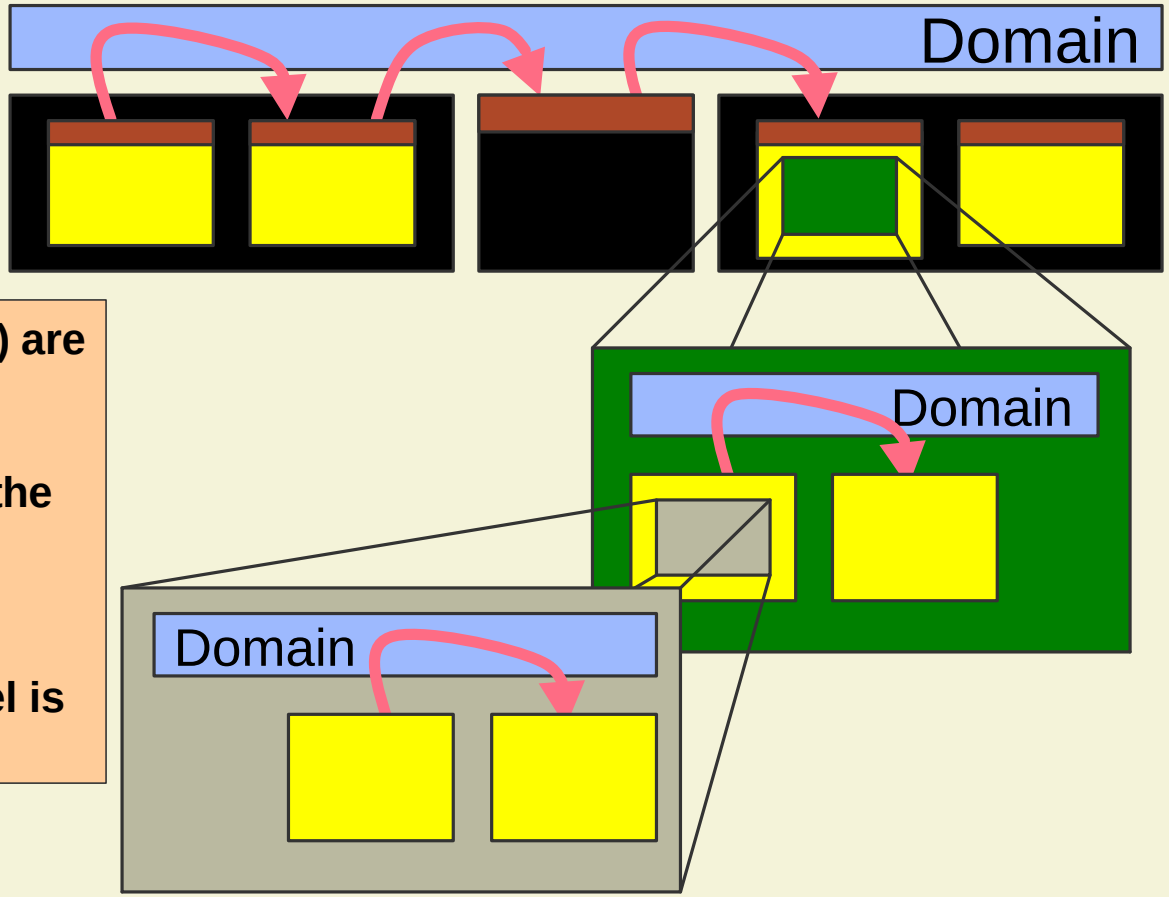


Problems:

- **With priority-based preemptive scheduling, how can you get precise mode changes?**
- **Given a validated design for a subsystem, how do you insert it in a system without invalidating it?**

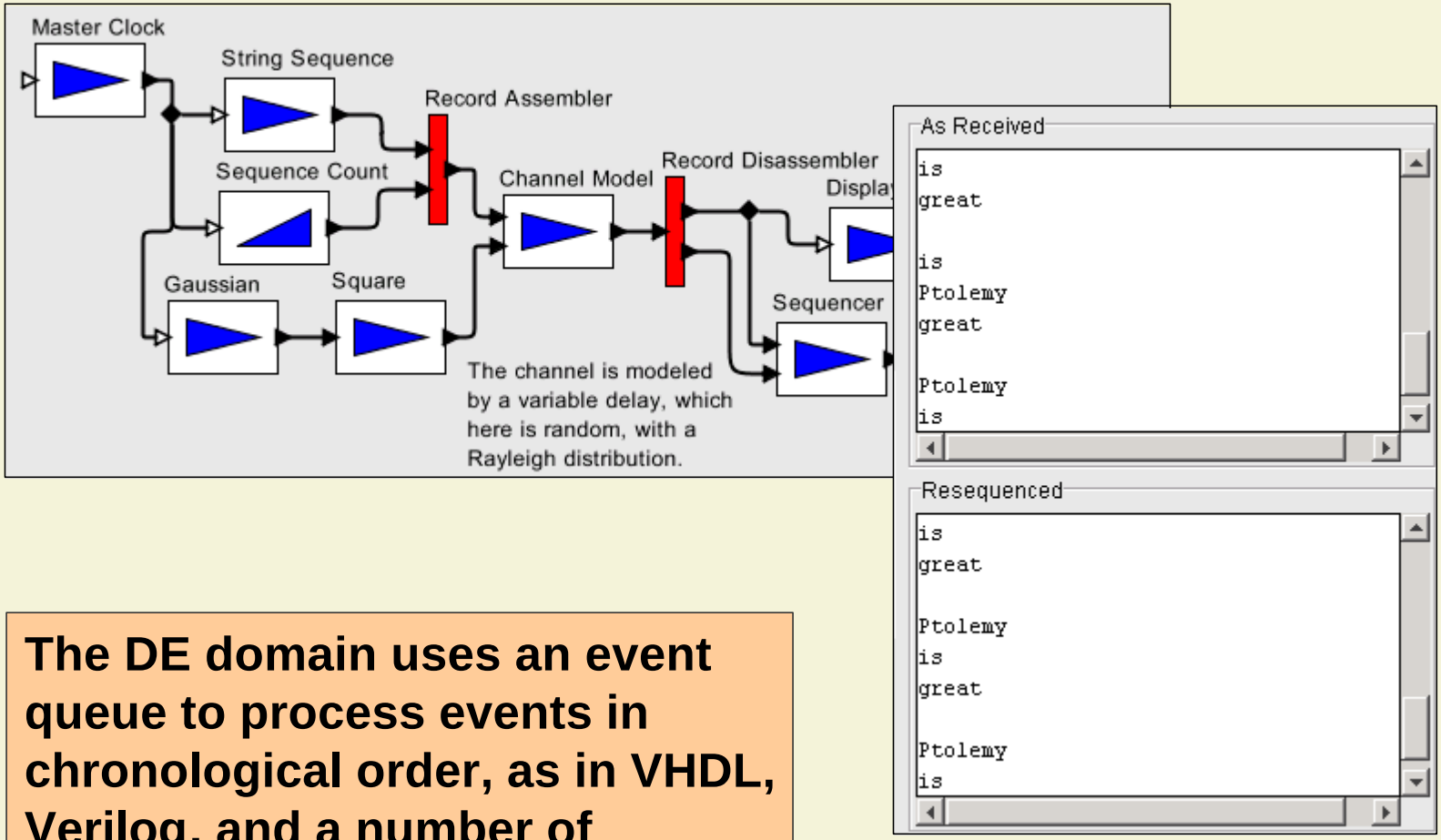
This approach is not compositional!

Hierarchical, Compositional Models



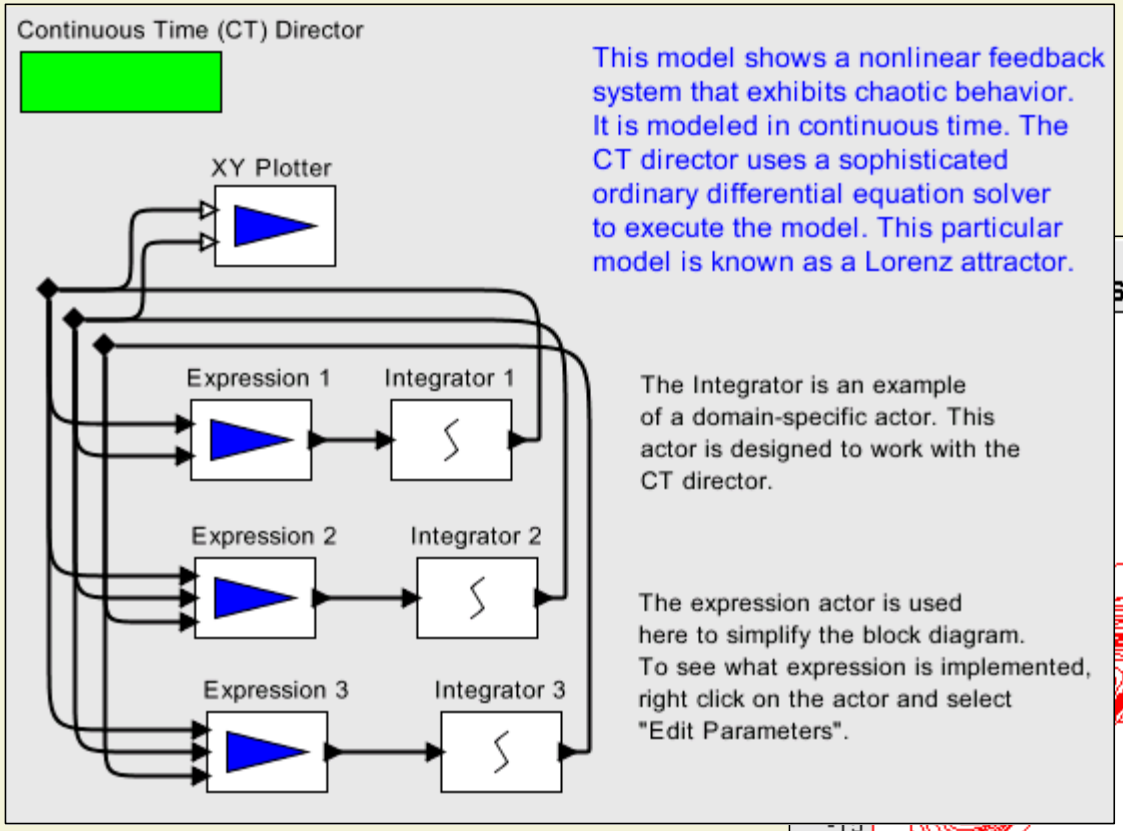
Schedulers (Directors) are nested hierarchically, each interacting with components through the Executable interface. Directors themselves implement this same interface, so the model is compositional.

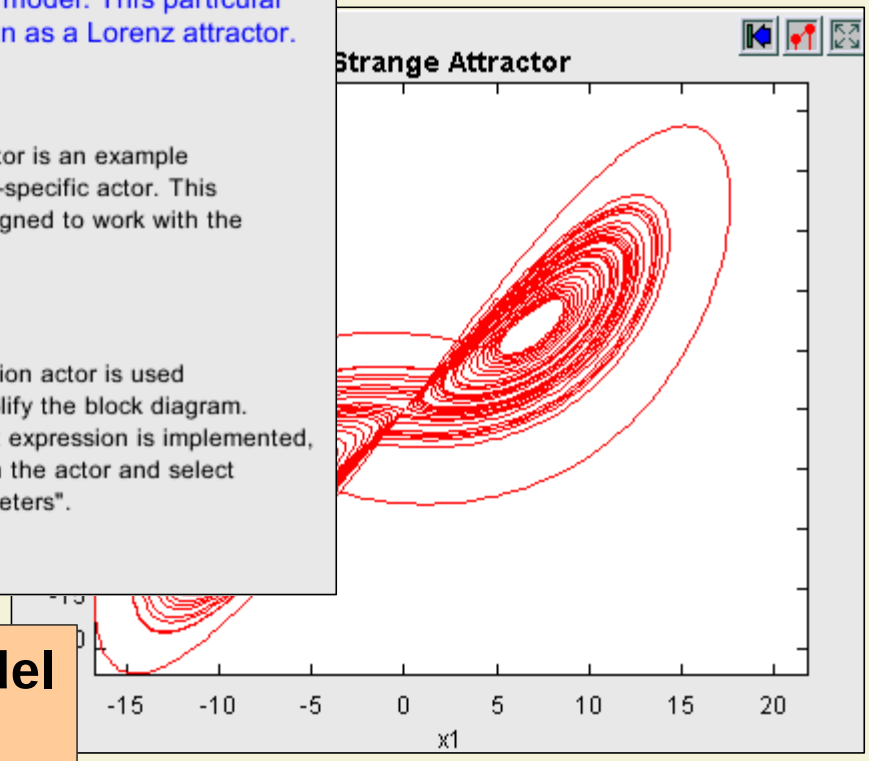
More Mature Domain 1: Discrete Events



The DE domain uses an event queue to process events in chronological order, as in VHDL, Verilog, and a number of network simulation languages.

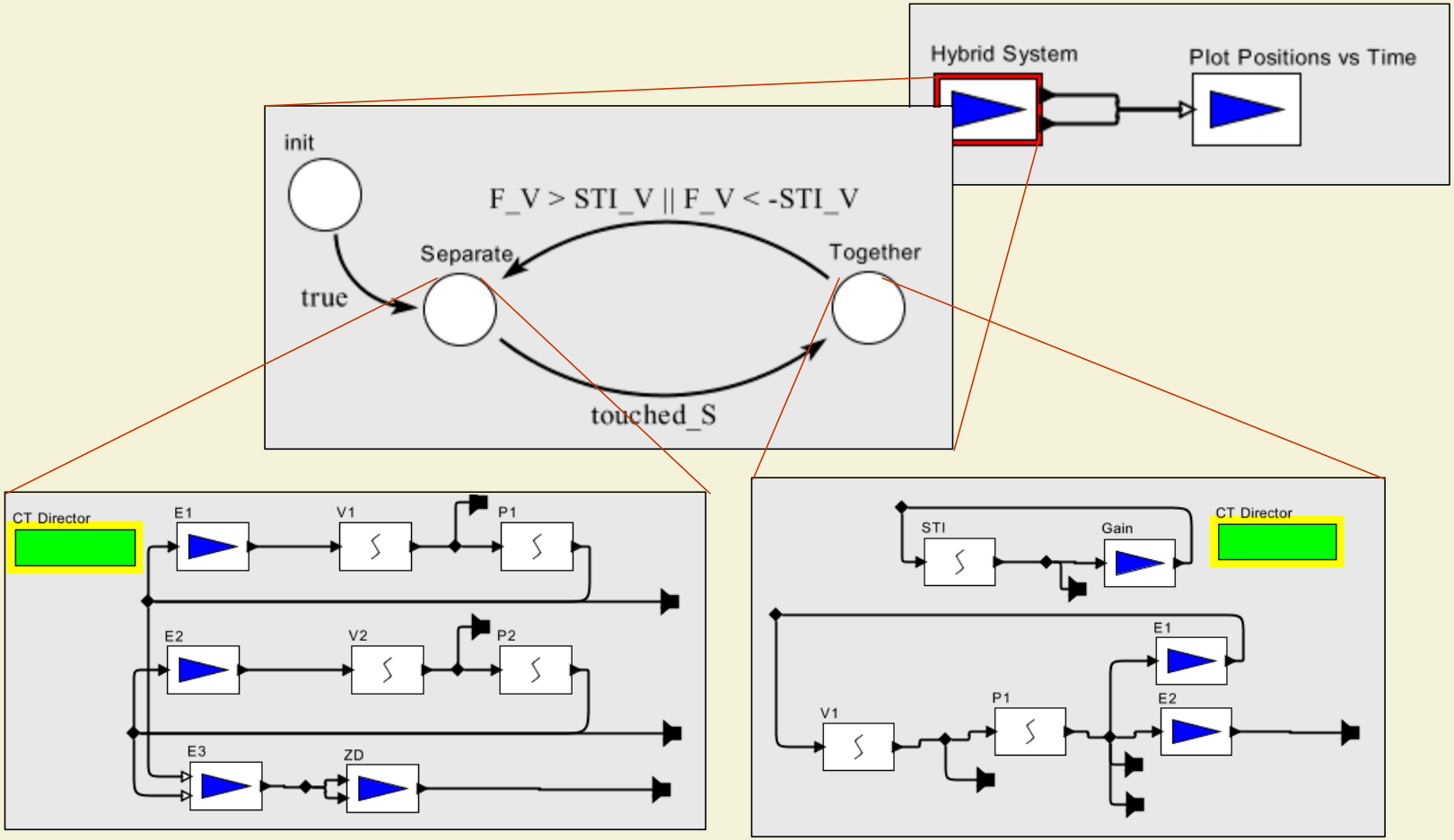
More Mature Domain 2: Continuous Time



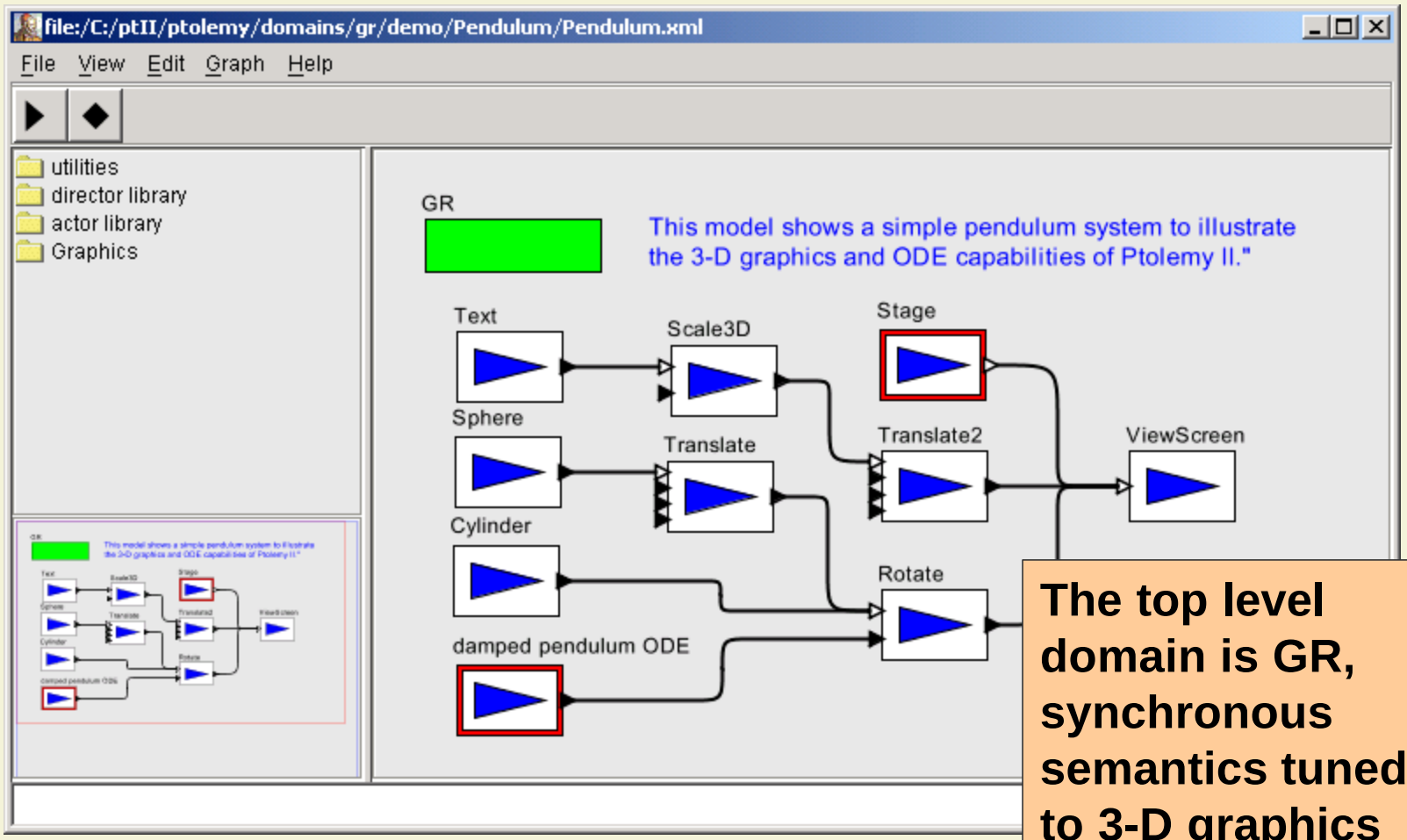


CT uses an ODE solver to model continuous-time systems.

Hybrid System Models: CT + FSM

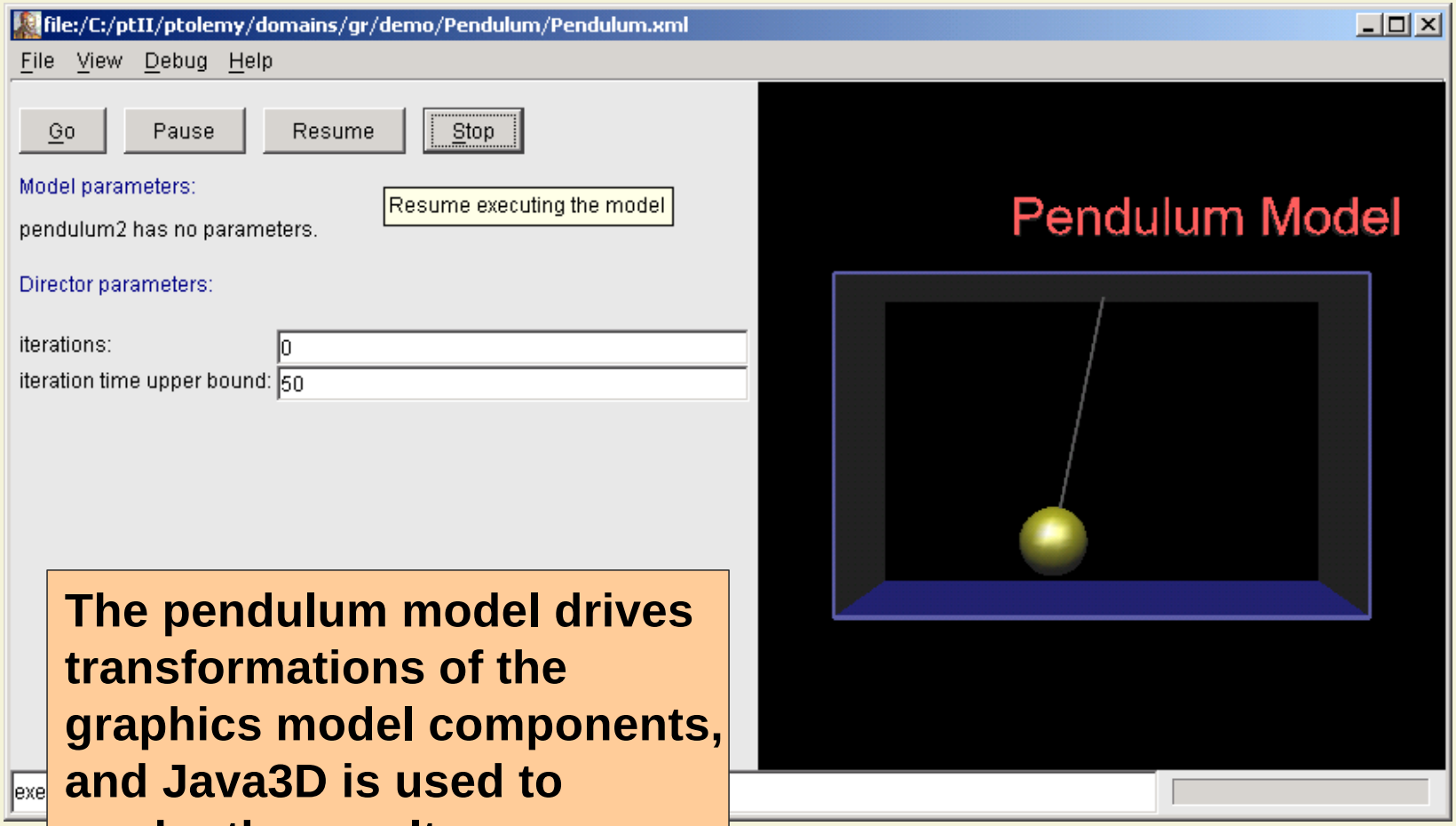


Experimental Domain-Specific Domain: GR, for 3D Graphics



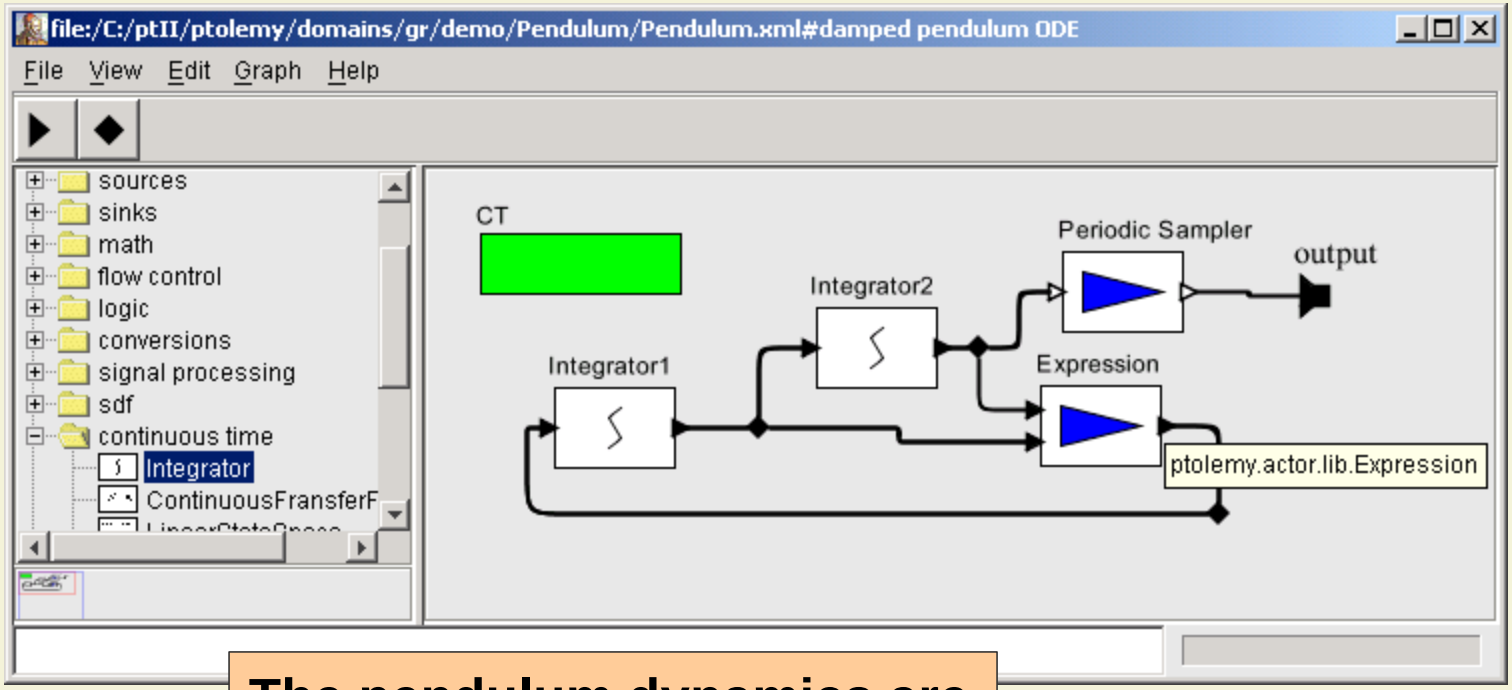
The top level domain is GR, synchronous semantics tuned to 3-D graphics

Animated Pendulum Model



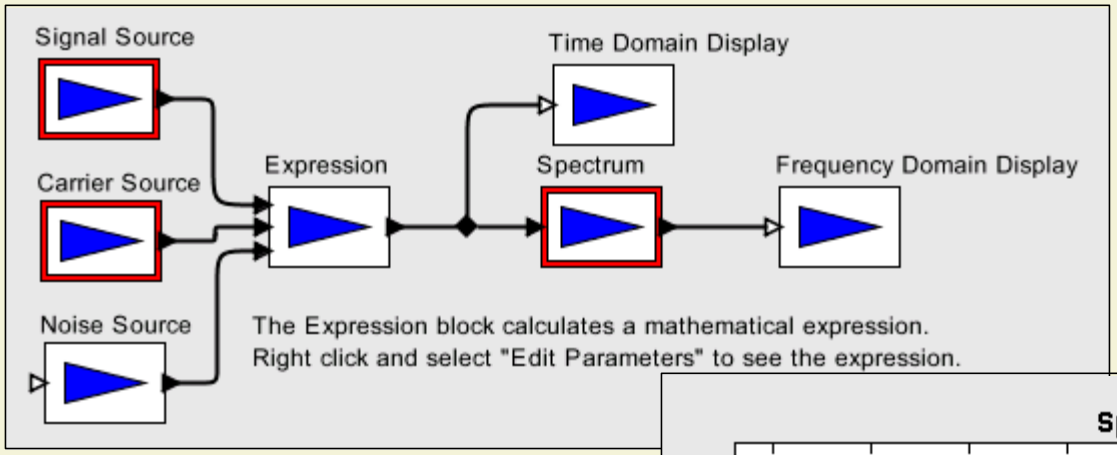
The pendulum model drives transformations of the graphics model components, and Java3D is used to render the result.

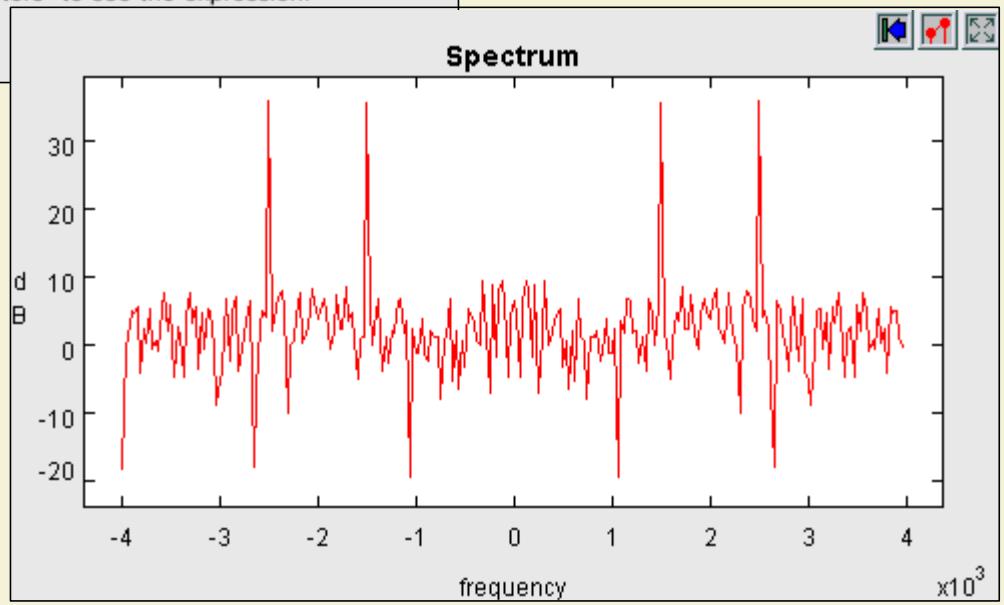
Physical Dynamics Realized in CT



The pendulum dynamics are modeled in the CT domain, which uses an ODE solver, as in Simulink.

More Mature Domains 3: Synchronous Dataflow

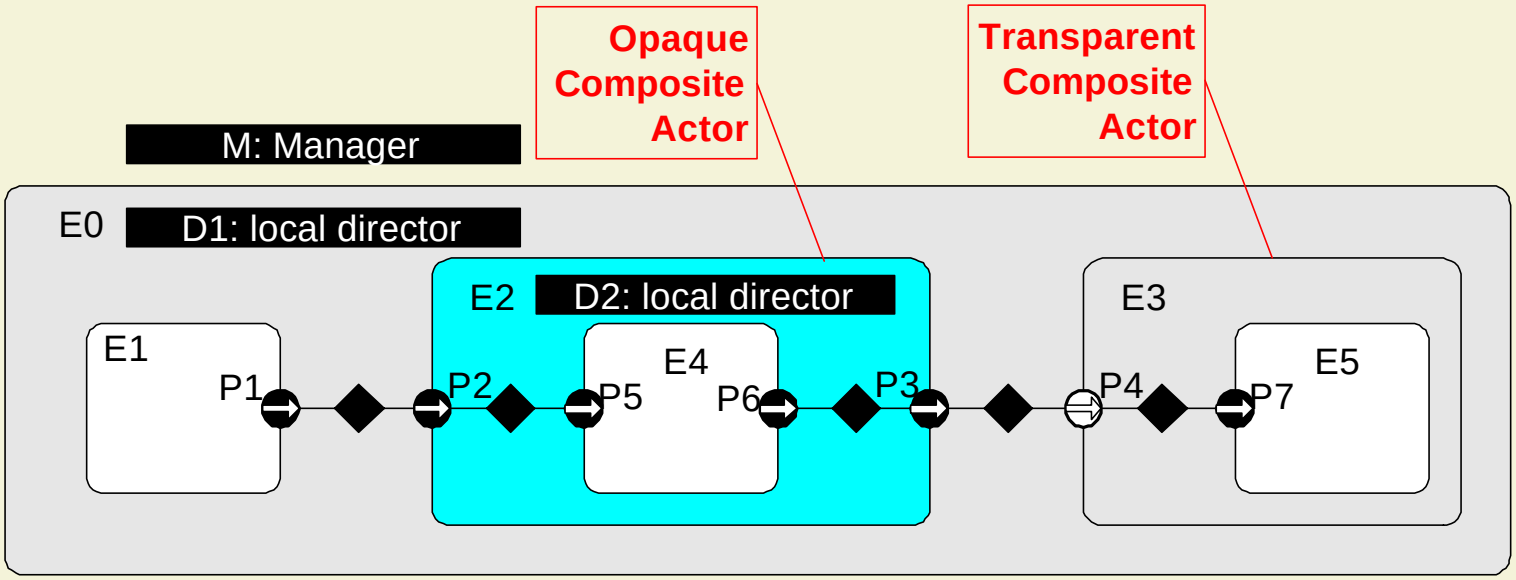




The SDF domain does static dataflow analysis to construct compile-time schedules, and analyze for deadlock and bounded memory.

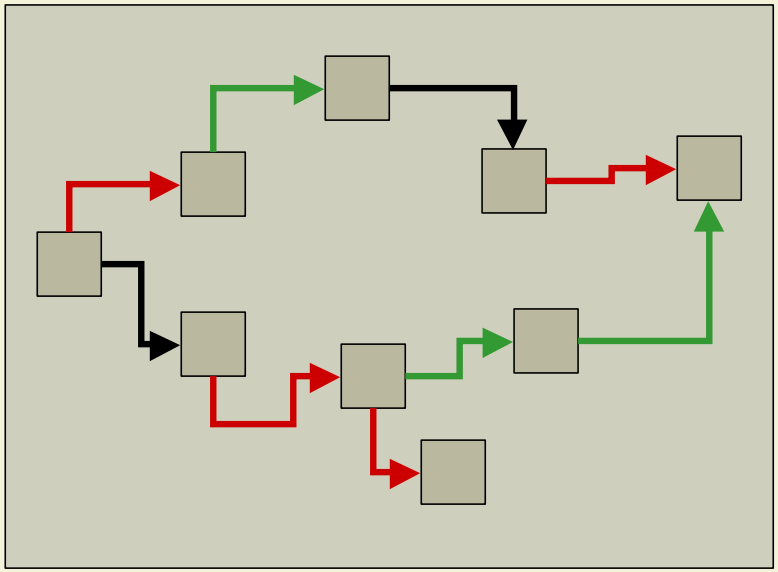
Hierarchical Heterogeneity and the Concept of "Domain"

Directors are domain-specific. A composite actor with a director becomes opaque. The Manager is domain-independent.



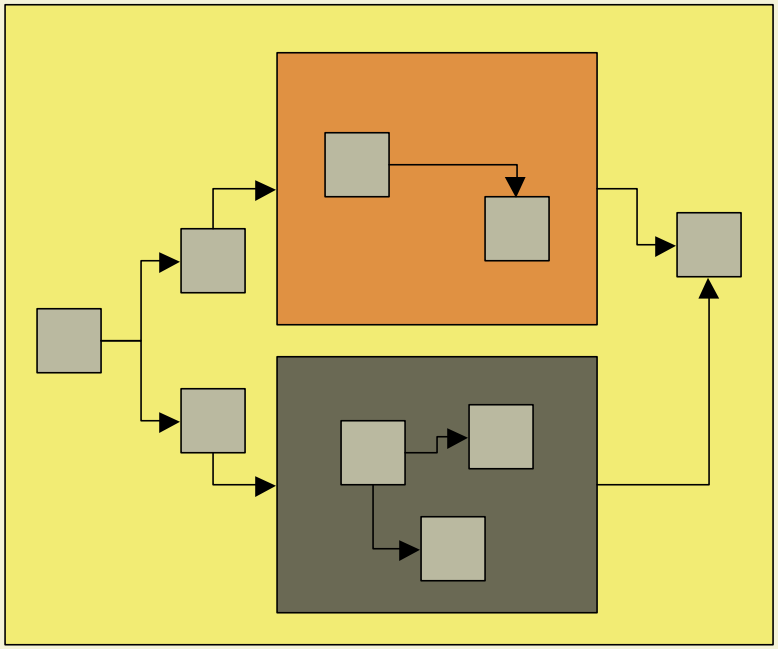
Hierarchical Heterogeneity vs. Amorphous Heterogeneity

Amorphous



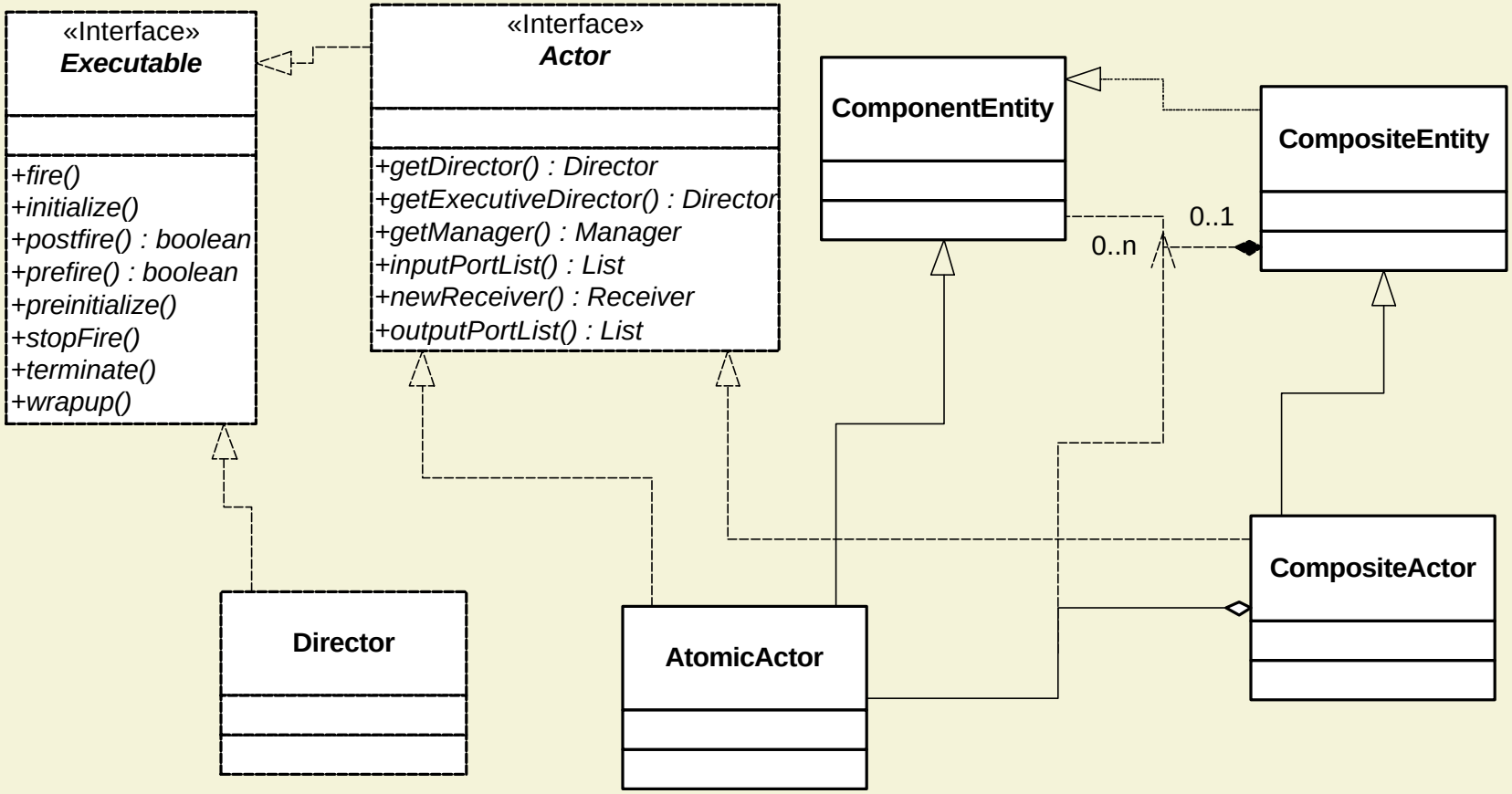
Color is a communication protocol only, which interacts in unpredictable ways with the flow of control.

Hierarchical



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

Basic Object Model for Executable Components

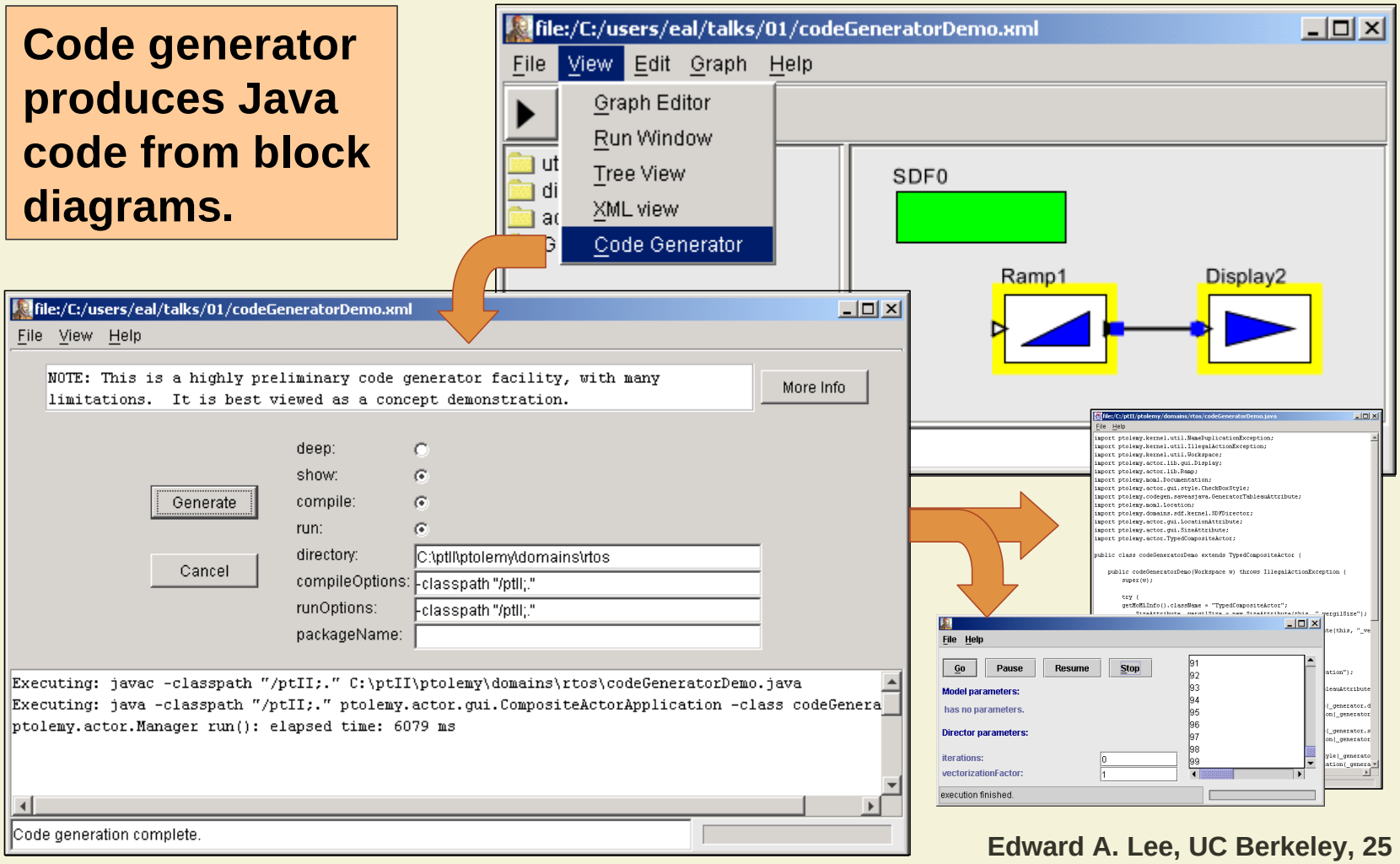


Contrast with Current Practice

- **Processes or threads under RTOS control**
  - two or three level hierarchy
  - limited control over flow of control (priorities)
  - not compositional
- **UML notations for concurrency**
  - Active objects (threads, not compositional)
  - Sequence diagrams (not scalable nor compositional)
  - Statecharts (specialized synchronous concurrency)

Preliminary Code Generator "Shallow" and "Deep"

Code generator produces Java code from block diagrams.



Near Term Plans

- Higher-order functions (with input from Yale & OGI)
- Code generator solidification
- System-level types codification
- Visual interface solidification (esp. for FSMs)
- Giotto semantics solidification
- Giotto code generator for TTA
- Hierarchical RTOS domain
- Publish and subscribe to CORBA Event Channel
  - (we have this for JavaSpaces)
- Synchronous/Reactive domain
- Dynamic Dataflow domain

Higher-Order Functions RFC

- Token whose value is a model
  - An actor with ports and parameters
- Parameter values can be models
- Apply actor with “model” parameter
- Apply actor with “model” input port
- Map combinator actor
- Zip combinator actor
- …

Our expectation is that these will vastly improve the expressiveness of visual syntaxes.