Overview of Ptolemy II

Edward A. Lee, PI
UC Berkeley

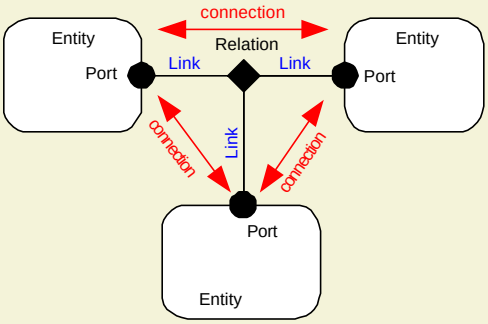
John Davis II, Elaine Cheong, Chamberlain Fong, Mudit Goel, Christopher Hylands, Jörn Janneck, Bart Kienhuis, Jie Liu, Xiaojun Liu, Lukito Muliadi, Stephen Neuendorffer, John Reekie, Sonia Sachs, Niel Smyth, Mary P. Stewart, Jeff Tsay, Brian K. Vogel, Paul Whitaker, Yuhong Xiong

Ptolemy Miniconference
Berkeley, CA, March 22-23, 2001

Status Update

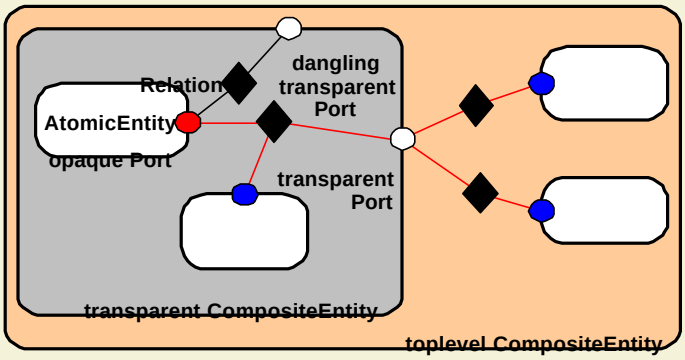
- Ptolemy II version 1.0 available
  - A platform for experimentation, collaboration
  - Open source, open architecture
  - Rated code (red, yellow, green)
  - Core code is very high quality (green)
  - Code is written to be read
  - Extensible GUI (all red, currently)
- Commercial support organization
  - recently formed: Agile Design

Components and their Relationships

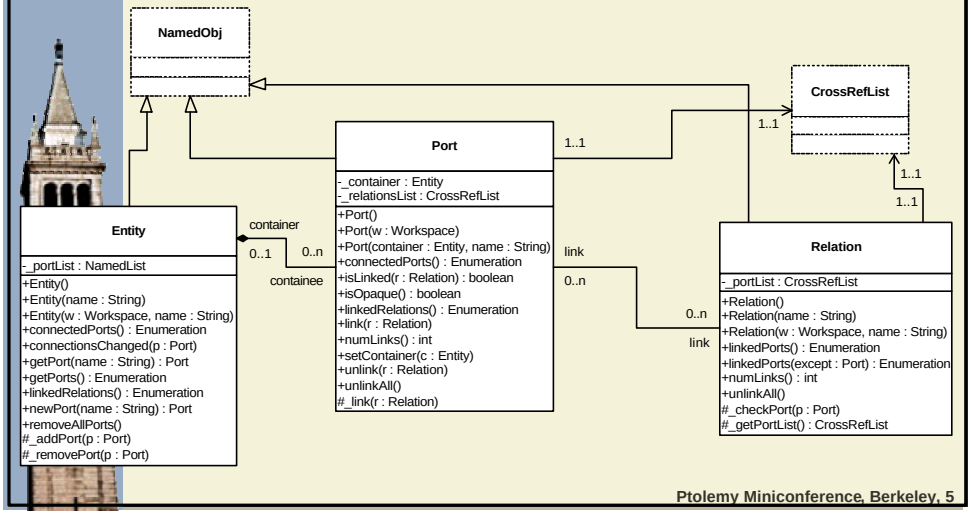


The Ptolemy II kernel provides an *abstract syntax* - clustered graphs - that is well suited to a wide variety of component-based modeling strategies, ranging from state machines to process networks.

Hierarchy - Construct components from finer grain components.

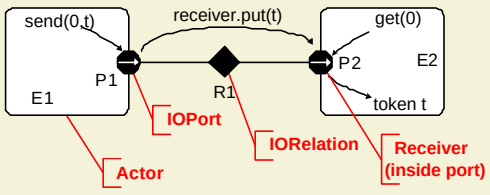


Basic Kernel Classes



Actor Package – Infrastructure for Producer/Consumer Components

Basic Transport:



Services in the Infrastructure:

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

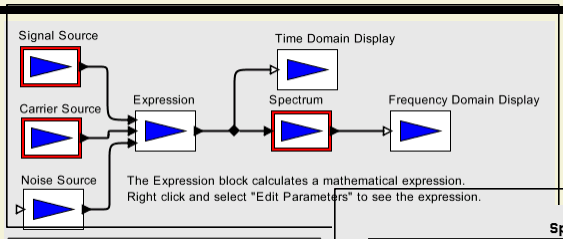
Domains – Provide semantic models for component interactions

- 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 *
- GR – 3-D graphics *
- PN – process networks
- RTOS – priority-driven reactive models *
- SDF – synchronous dataflow
- SR – synchronous/reactive **

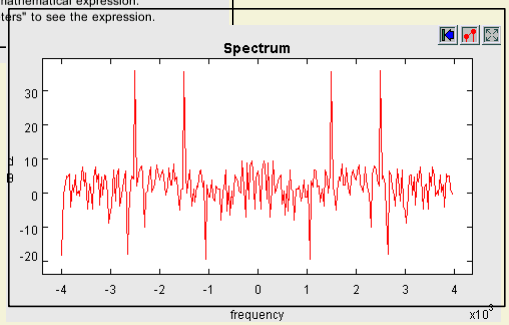
* New domains in Ptolemy II
** Not yet realized in Ptolemy II

SDF domain author:
Steve Neuendorffer

Synchronous Dataflow

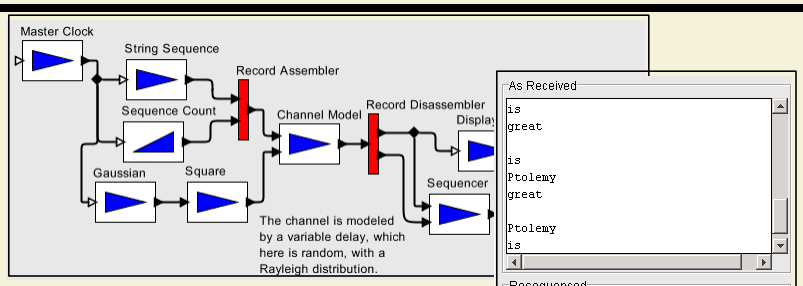


The SDF domain does static dataflow analysis to construct compile-time schedules, and analyze for deadlock and bounded memory.



DE domain authors:
Lukito Muliadi
Jie Liu

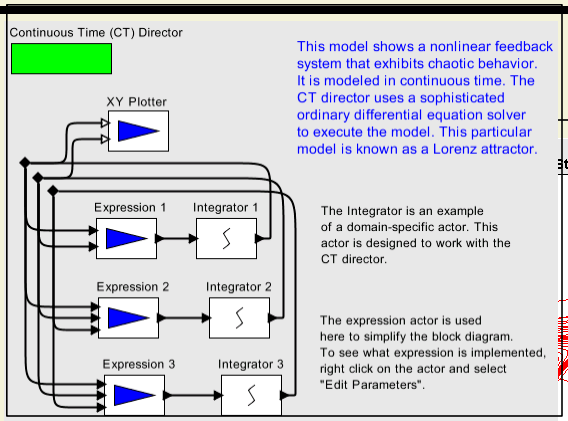
Discrete Event Models

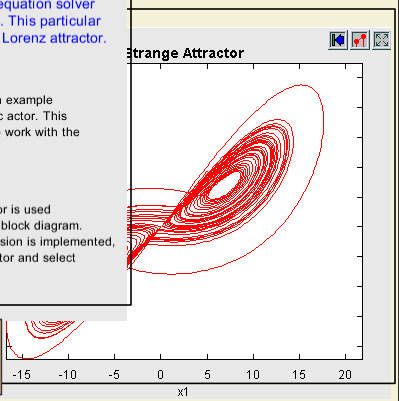


The DE domain uses an event queue to process events in chronological order, as in VHDL, Verilog, and a number of network simulation languages.

CT domain author:
Jie Liu

Continuous Time Models



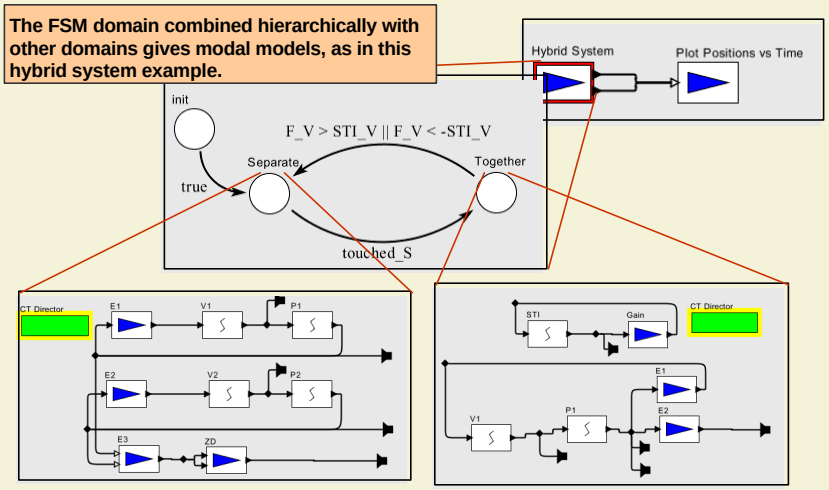


CT uses an ODE solver to model continuous-time systems.

FSM domain author:
Xiaojun Liu

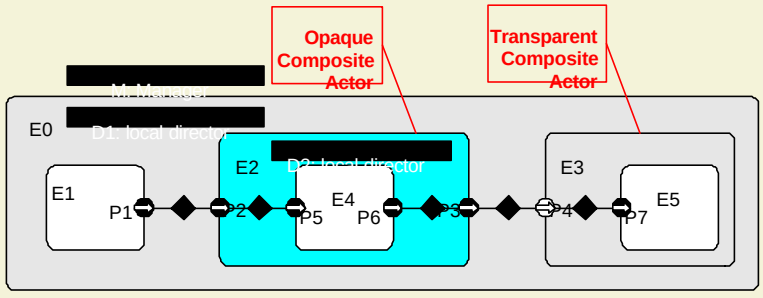
Finite State Machines

The FSM domain combined hierarchically with other domains gives modal models, as in this hybrid system example.

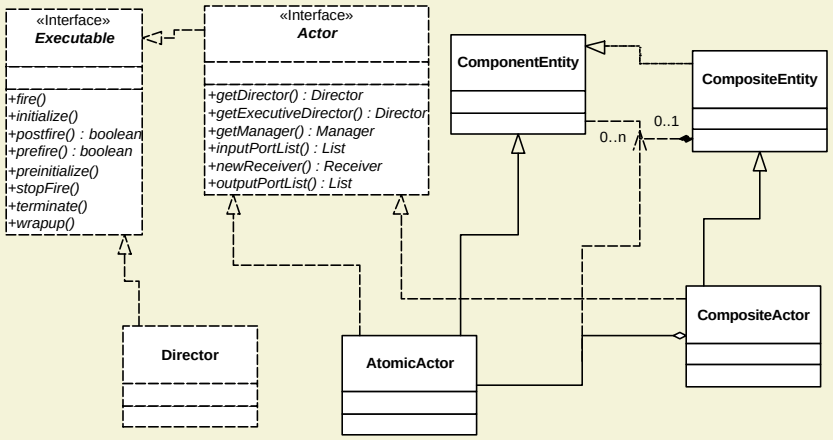


Hierarchical Heterogeneity

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



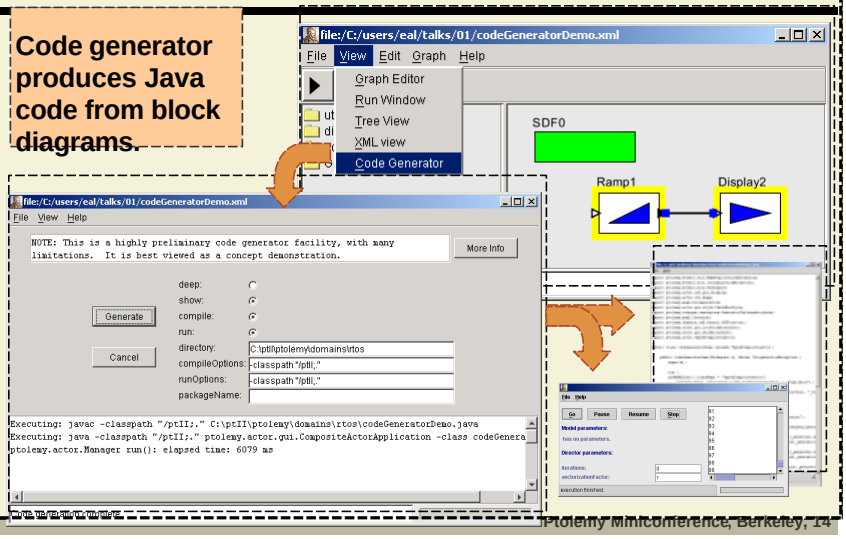
Basic Object Model for Executable Components



Code Generator: "Shallow" and "Deep"

Code gen author: Jeff Tsay

Code generator produces Java code from block diagrams.

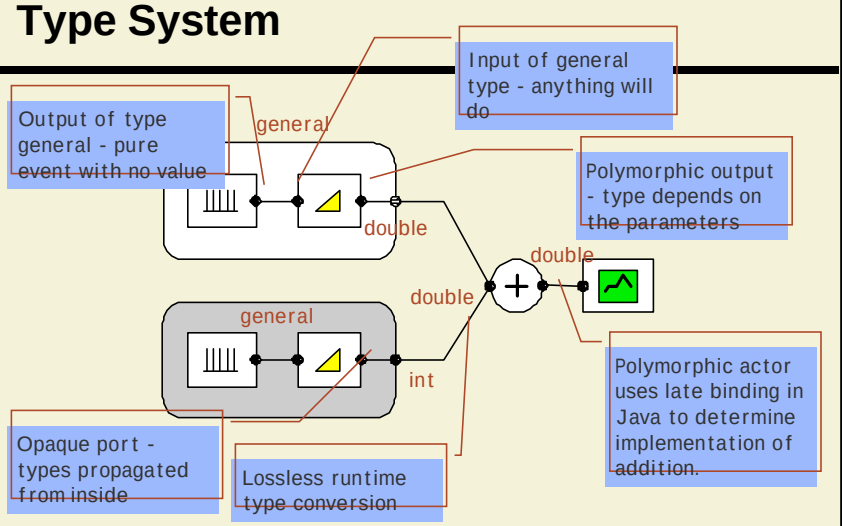


Infrastructure Support

- **Vergil – visual editor** (Steve Neuendorffer)
- **MoML – XML schema** (Edward Lee, Steve Neuendorffer)
- **Expression language** (Neil Smyth, Xiaojun Liu)
- **Type system** (Yuhong Xiong)
- **Math package** (Jeff Tsay, Bart Kienhuis, William Wu)
- **Graph package** (Yuhong Xiong)
- **Plot package** (Edward Lee, Christopher Hylands)
- **3-D graphics** (Chamberlain Fong)
- **Actor library** (all)

Type System

Type system author: Yuhong Xiong



Software Practice

- Object models in UML
- Design patterns
- Layered software architecture
- Design and code reviews
- Design document
- Nightly build
- Regression tests
- Sandbox experimentation
- Code rating

Code rating

- A simple framework for
  - quality improvement by peer review
  - change control by improved visibility
- Four confidence levels
  - **Red.** No confidence at all.
  - **Yellow.** Passed design review. Soundness of the APIs.
  - **Green.** Passed code review. Quality of implementation.
  - **Blue.** Passed final review. Backwards-compatibility assurance.