Ptolemy Project Status and Overview

Edward A. Lee

Ptolemy Project Director, UC Berkeley

6th Biennial Ptolemy Miniconference

Berkeley, CA
May 12, 2005

Software Legacy of the Project

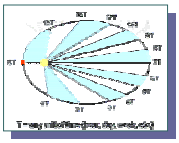
- Gabriel (1986-1991)
  - Written in Lisp
  - Aimed at signal processing
  - Synchronous dataflow (SDF) block diagrams
  - Parallel schedulers
  - Code generators for DSPs
  - Hardware/software co-simulators
- Ptolemy Classic (1990-1997)
  - Written in C++
  - Abstract Actor Semantics
  - Multiple models of computation
  - Hierarchical heterogeneity
  - Dataflow variants: BDF, DDF, PN
  - C/VHDL/DSP code generators
  - Optimizing SDF schedulers
  - Higher-order components
- Ptolemy II (1996-2022)
  - Written in Java
  - Behavioral polymorphism
  - Multithreaded
  - Network integrated and distributed
  - Modal models
  - Sophisticated type system
  - CT, HDF, CI, GR, etc.

Each of these served us, first-and-foremost, as a laboratory for investigating design.

Focus has always been on embedded software.

And Most Recently…

- Kepler (2003-?)
  - Scientific workflows
  - Web services harvester
  - Computational grid integration
  - Semantic types
  - Browser interface
  - Database integration
  - "R" integration
  - Sensor data streaming
  - XML and XSLT integration
  - …



ROADNet

SEEK

GEON
The Geosciences Network

BIRN
Biomedical Informatics Research Network

SciDAC
Scientific Discovery through Advanced Computing

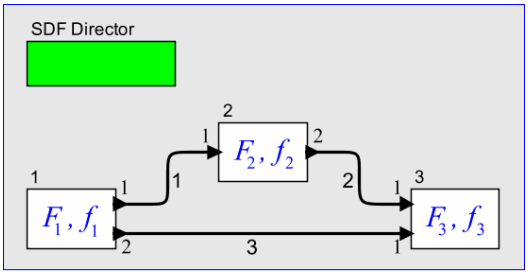
SDM

DARPA

NSF

Lee, Berkeley 3

Where it started: SDF: Synchronous Dataflow and the Balance Equations (1985-86)



production/consumption matrix

Connector 1

$$\Gamma = \begin{bmatrix} 1 & -1 & 0 \\ 0 & 2 & -1 \\ 2 & 0 & -1 \end{bmatrix}$$

Actor 1

$$q = \begin{bmatrix} q_1 \\ q_2 \\ q_3 \end{bmatrix}$$

firing vector

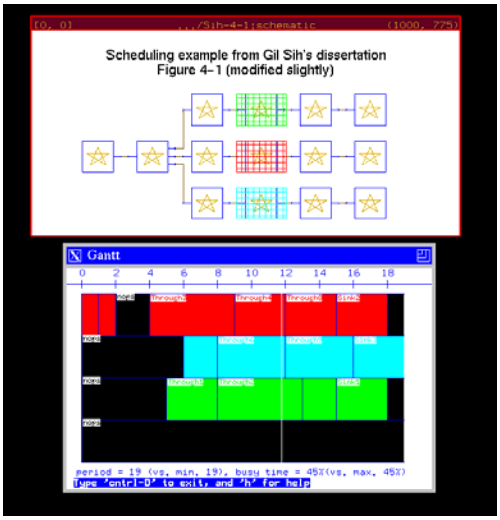
balance equations

$$\Gamma q = \vec{0} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

Lee, Berkeley 4

Gabriel and Ptolemy Classic Leveraged SDF to Generate Parallel Code

SDF model, parallel schedule, and synthesized DSP assembly code (1990)

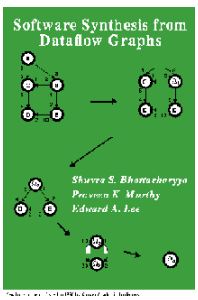



```
codeblock(std) {
; initialize address registers for coef and
delayLineove     #$addr(coef)+$val(coefLen)-1,r3
; insert here
        move    $ref(delayLineStart),r5
; delayLine
        move    #$val(stepSize),x1
        move    $ref(error),x0
        mpyr    x0,x1,a
        move    a,x0
        move    x:(r3),b        y:(r5)+,y0
        }

        codeblock(loop) {
        do      #$val(loopVal),$label(endloop)
        macr    x0,y0,b
        move    b,x:(r3)-
        move    x:(r3),b        y:(r5)+,y0
$label(endloop)
        }

        codeblock(noloop) {
        macr    x0,y0,b
        move    b,x:(r3)-
        move    x:(r3),b        y:(r5)+,y0
        }
```

It is an interesting (and rich) research problem to minimize interlocks in complex multirate applications.

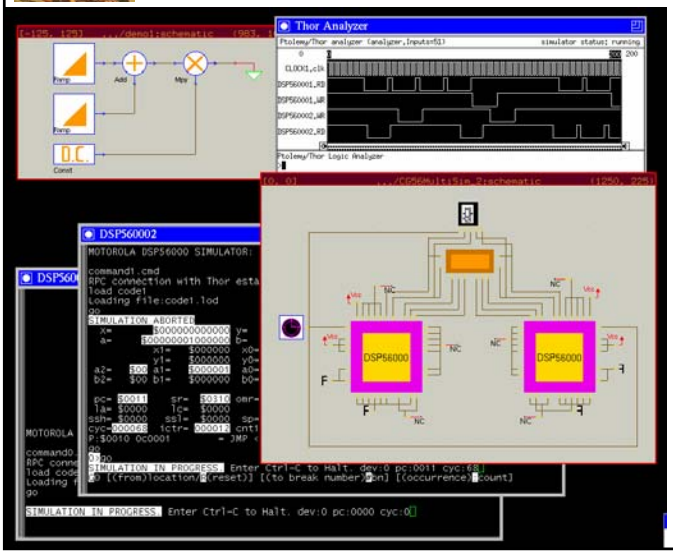
Many Scheduling and Optimization Problems (and Some Solutions) Resulted

- Optimization criteria that might be applied:
- Minimize buffer sizes.
- Minimize the number of actor activations.
- Minimize the size of the representation of the schedule (code size).



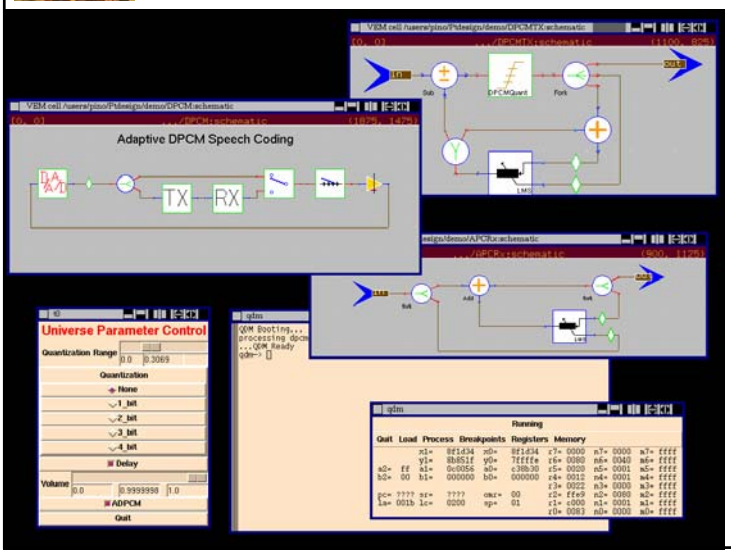
See S. S. Bhattacharyya, P. K. Murthy, and E. A. Lee, *Software Synthesis from Dataflow Graphs,* Kluwer Academic Press, 1996, for a summary of the single processor optimization problems.

Gabriel and Ptolemy Classic Provided Cosimulation of Hardware and Generated Software



An SDF model, a "Thor" model of a 2-DSP architecture, a "logic analyzer" trace of the execution of the architecture, and two DSP code debugger windows, one for each processor (1990).

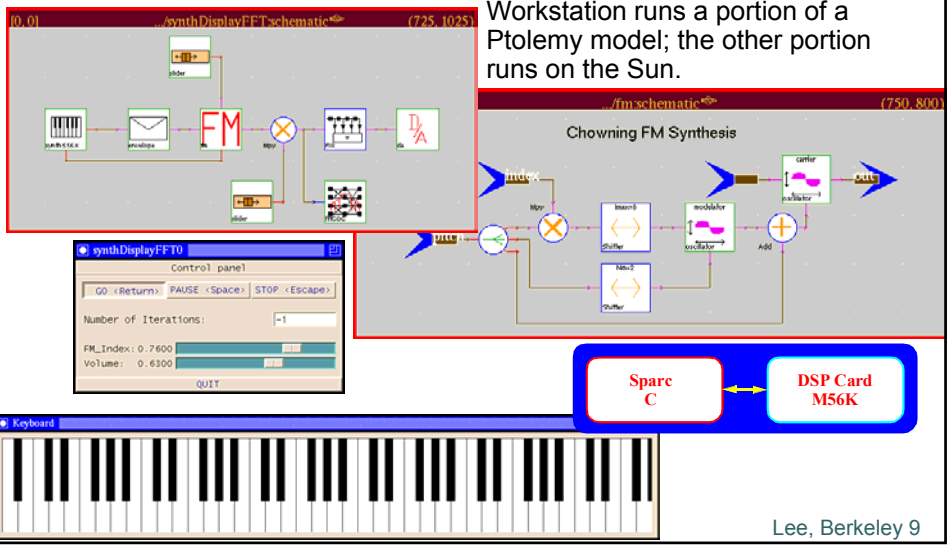
Example: Application to ADPCM Speech Coding



Model of a speech coder generated to DSP assembly code and executed using aa DSP debugger interface with host/DSP interaction (1993).

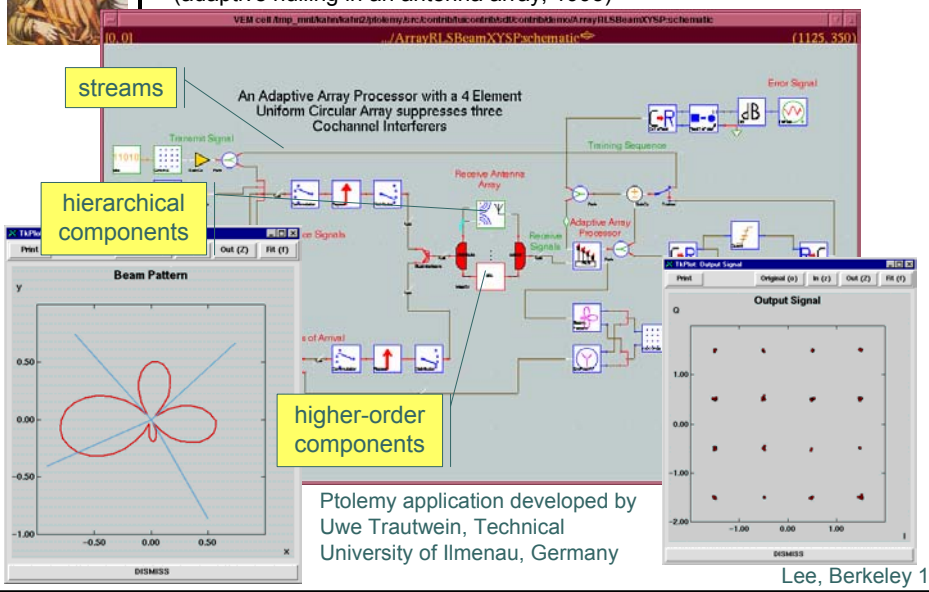
Example: Heterogeneous Architecture with DSP and Sun Sparc Workstation (1995)

DSP card in a Sun Sparc Workstation runs a portion of a Ptolemy model; the other portion runs on the Sun.



Ptolemy Classic Example Showing Higher-Order Components

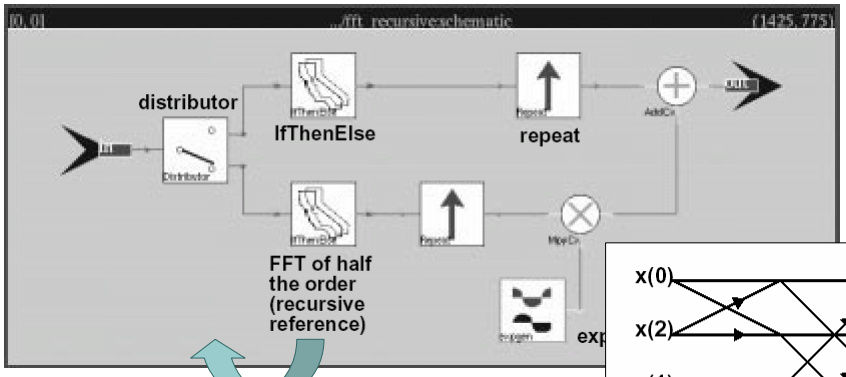
(adaptive nulling in an antenna array, 1995)



Ptolemy application developed by Uwe Trautwein, Technical University of Ilmenau, Germany

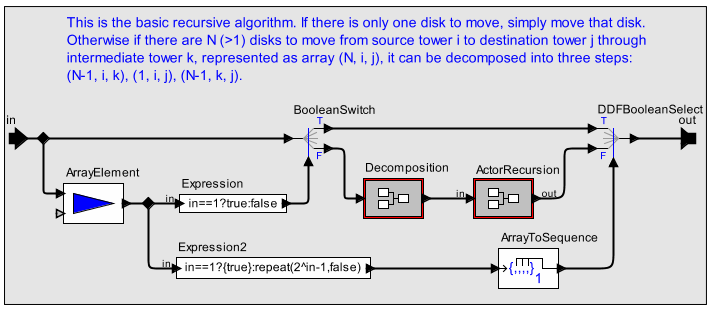
Higher-Order Components Realizing Recursion in Ptolemy Classic

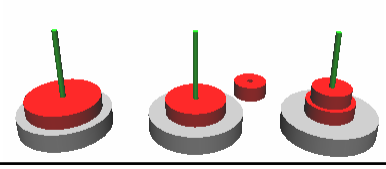
FFT implementation in Ptolemy Classic (1995) used a partial evaluation strategy on higher-order components.



Higher-Order Components in Ptolemy II

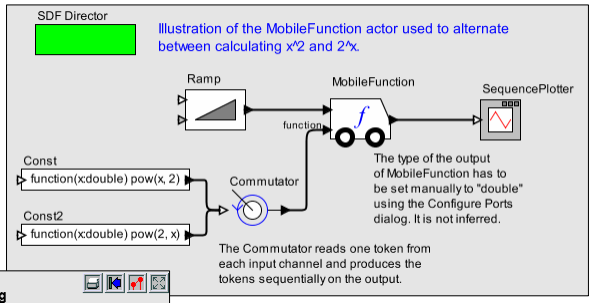
The dynamic dataflow (DDF) domain (new to Ptolemy II in v. 5.0) implements recursion in a similar way [due to Gang Zhou].

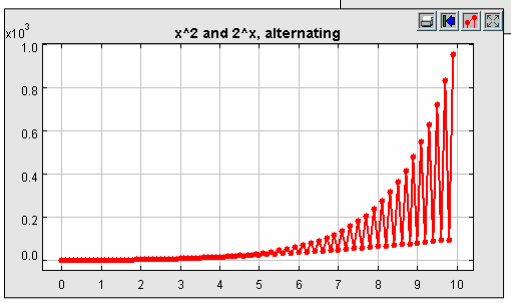




Higher-Order Expression Language in Ptolemy II

Higher-order components (actor-oriented) coupled with a higher-order expressions (functional) are a potentially powerful combination.



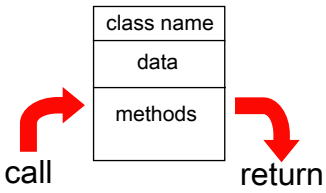


Neuendorffer and Zhao

What is the Ptolemy Project Really About?

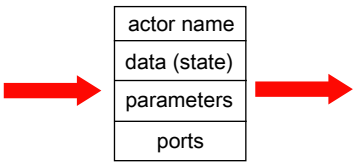
The Ptolemy Project is About Actor-Oriented Design

Object orientation:



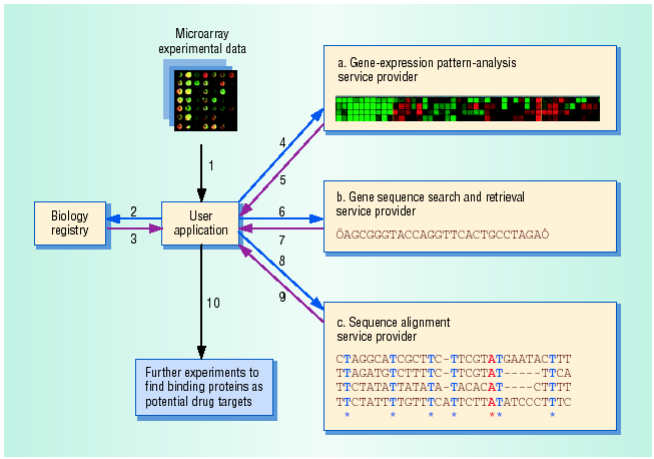
What flows through an object is sequential control

Actor orientation:



What flows through an object is streams of data

Actor-Oriented vs. Object-Oriented

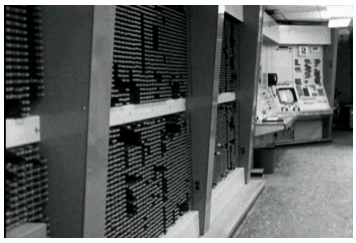


From Gao & Hayes, "Integrating Biological Research through Web Services," *Computer*, March, 2005.

The figure at the left shows the use of object-oriented web services for a "microarray data-analysis scenario for identifying targets in drug discovery." The authors explain, "the numbered lines are the steps in the analysis path."

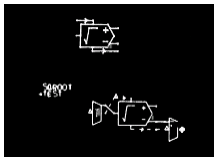
The First (?) Actor-Oriented Platform

The On-Line Graphical Specification of Computer Procedures
W. R. Sutherland, Ph.D. Thesis, MIT, 1966



MIT Lincoln Labs TX-2 Computer

Bert Sutherland with a light pen



Bert Sutherland used the first acknowledged object-oriented framework (Sketchpad, created by his brother, Ivan Sutherland) to create the first actor-oriented programming framework.

Partially constructed actor-oriented model with a class definition (top) and instance (below).

Your Speaker in 1966

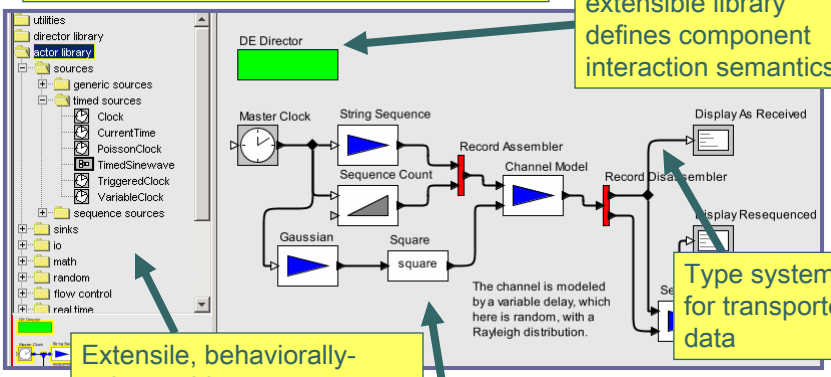
Modern Examples of Actor-Oriented Platforms

- Simulink (The MathWorks)
- LabVIEW (National Instruments)
- Modelica (Linkoping)
- OPNET (Opnet Technologies)
- Giotto and xGiotto (UC Berkeley)
- Polis & Metropolis (UC Berkeley)
- Gabriel, Ptolemy, and Ptolemy II (UC Berkeley)
- OCP, open control platform (Boeing)
- GME, actor-oriented meta-modeling (Vanderbilt)
- SPW, signal processing worksystem (Cadence)
- System studio (Synopsys)
- ROOM, real-time object-oriented modeling (Rational)
- Easy5 (Boeing)
- Port-based objects (U of Maryland)
- I/O automata (MIT)
- VHDL, Verilog, SystemC (Various)
- …

Ptolemy II: Our Laboratory for Actor-Oriented Models of Computation

Concurrency management supporting dynamic model structure.

Director from an extensible library defines component interaction semantics



Type system for transported data

Extensile, behaviorally-polymorphic component library.

Visual editor supporting an abstract syntax

Models of Computation Implemented in Ptolemy II

- CI – Push/pull component interaction
- Click – Push/pull with method invocation
- CSP – concurrent threads with rendezvous
- CT – continuous-time modeling
- DDF – Dynamic dataflow
- DE – discrete-event systems
- DDE – distributed discrete events
- DPN – distributed process networks
- FSM – finite state machines
- DT – discrete time (cycle driven)
- Giotto – synchronous periodic
- GR – 2-D and 3-D graphics
- PN – process networks
- SDF – synchronous dataflow
- SR – synchronous/reactive
- TM – timed multitasking

Most of these are actor oriented.

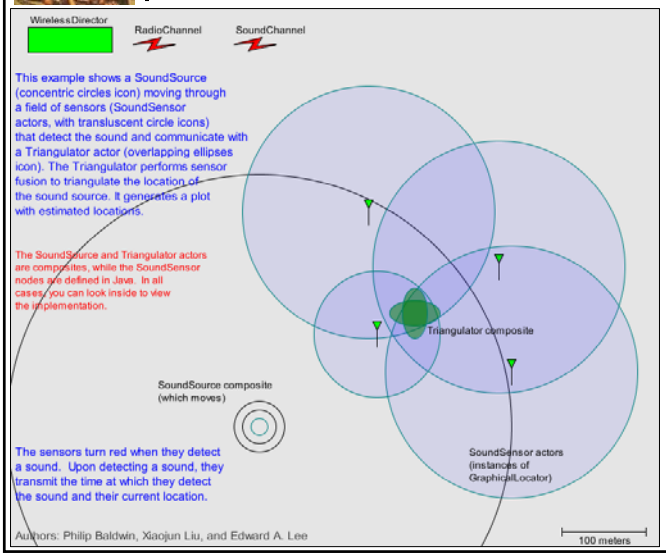
Ptolemy II Extension Points

- Define actors
- Interface to foreign tools (e.g. Python, MATLAB)
- Interface to verification tools (e.g. Chic)
- Define actor definition languages
- Define directors (and models of computation)
- Define visual editors
- Define textual syntaxes and editors
- Packaged, branded configurations

"Domains" are extensions built on the core infrastructure.

What have we done recently?

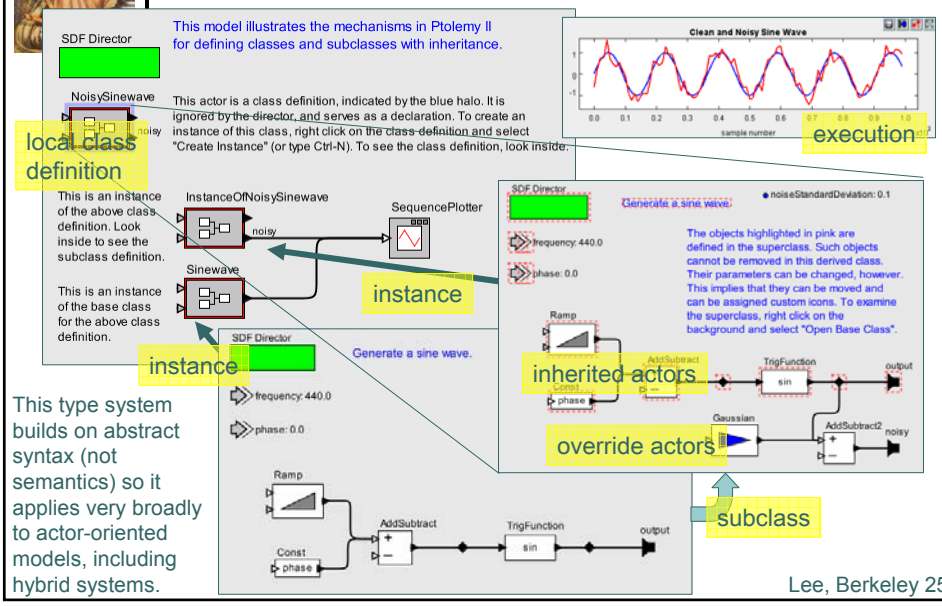
Modeling and Design of Wireless Networked Systems



VisualSense: Modeling of wireless sensor networks as an extension of DE. [Baldwin, Kohli, Liu, Zhao]

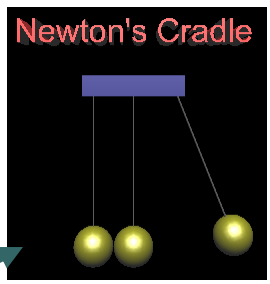
VIPTOS: Design of software for wireless sensor network motes in TinyOS/nesC. [Cheong, coming soon]

Actor-Oriented Type Systems
Classes, Subclasses, and Inheritance

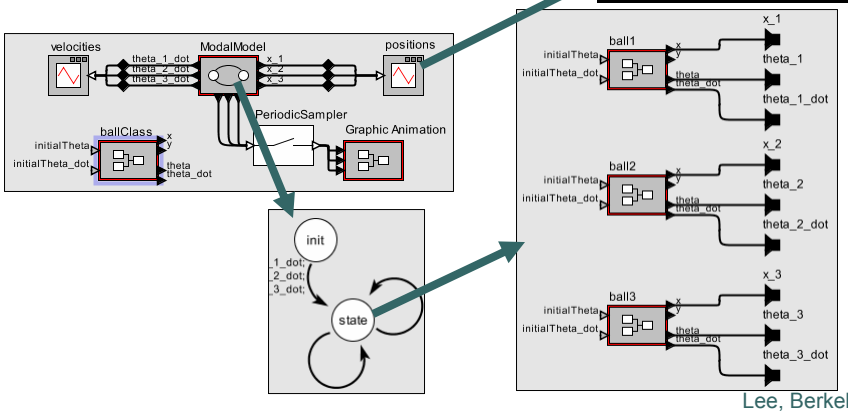


This type system builds on abstract syntax (not semantics) so it applies very broadly to actor-oriented models, including hybrid systems.

Semantics

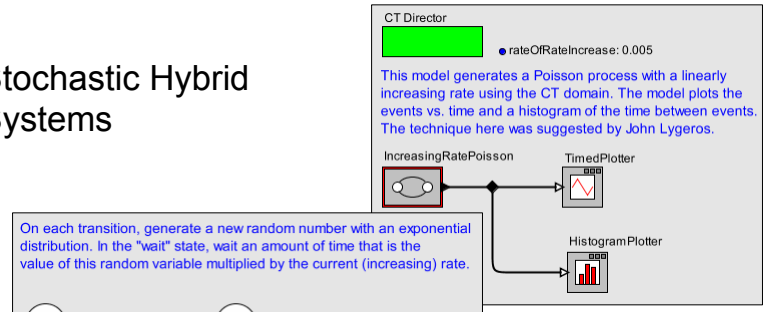


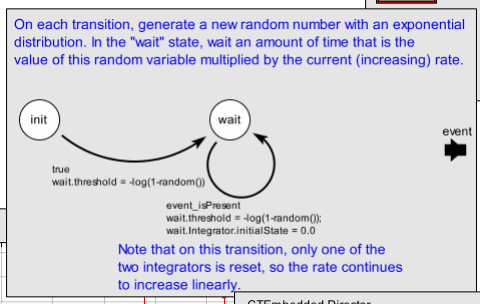
Clean, clear, and rigorous semantics for discrete-event, continuous-time, and hybrid systems [Cataldo, Liu, Matsikoudis, Zheng]



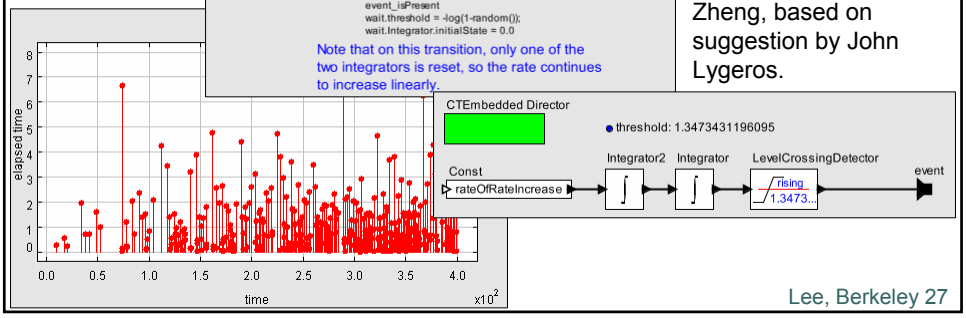
Stochastic Hybrid Systems

Stochastic hybrid systems in Ptolemy II are Monte-Carlo models of nondeterminism

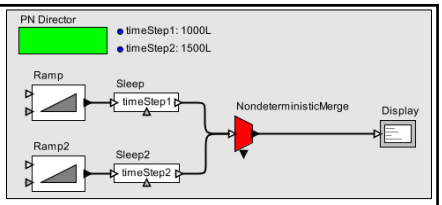




Example of random "spontaneous transitions" by Lee and Zheng, based on suggestion by John Lygeros.



Other Key Results



- Reconfiguration analysis [Neuendorffer]
- Generalized dependency analysis [Neuendorffer, Zheng]
- The Cal actor language [Eker and Janneck]
- Java code generation [Neuendorffer]
- Modal model semantics [Liu, Zhou (Rachel)]
- Mixed procedural and event semantics [Cheong]
- Unbounded time, controlled precision [Zheng]
- Nondeterministic merge in PN [Lee, Xiaowen Xin (LLNL)]
- Giotto + Ptolemy II package [Brooks]
- Communications library [Zhou (Rachel)]
- Image and video library [Yeh]
- Scratchpad memory management from SDF [Kohli]

Acknowledgements

- Current students
  - Adam Cataldo
  - Elaine Cheong
  - Thomas Huining Feng
  - Xiaojun Liu
  - Eleftherios Matsikoudis
  - Yang Zhao
  - Haiyang Zheng
  - Gang Zhou
  - Rachel Zhou
- Staff
  - Christopher Brooks
  - Mary Margaret Sprinkle
  - Mary Stewart
- Recent PhD graduates
  - Steve Neuendorffer (Xilinx)
  - Yuhong Xiong (HP Labs)
- Recent Postdocs
  - Jörn Janneck (Xilinx)
- Recent masters graduates
  - Vinay Krishnan
  - Sanjeev Kohli
  - James Yeh
- Current sponsors
  - Agilent
  - Hewlett-Packard
  - Escher Institute
  - National Science Foundation
  - Toyota