Ptutorial: Using and Extending Ptolemy II

Edward A. Lee
Robert S. Pepper Distinguished Professor and
Chair of EECS, UC Berkeley

EECS 249 Guest Lecture

Berkeley, CA
September 8, 2009

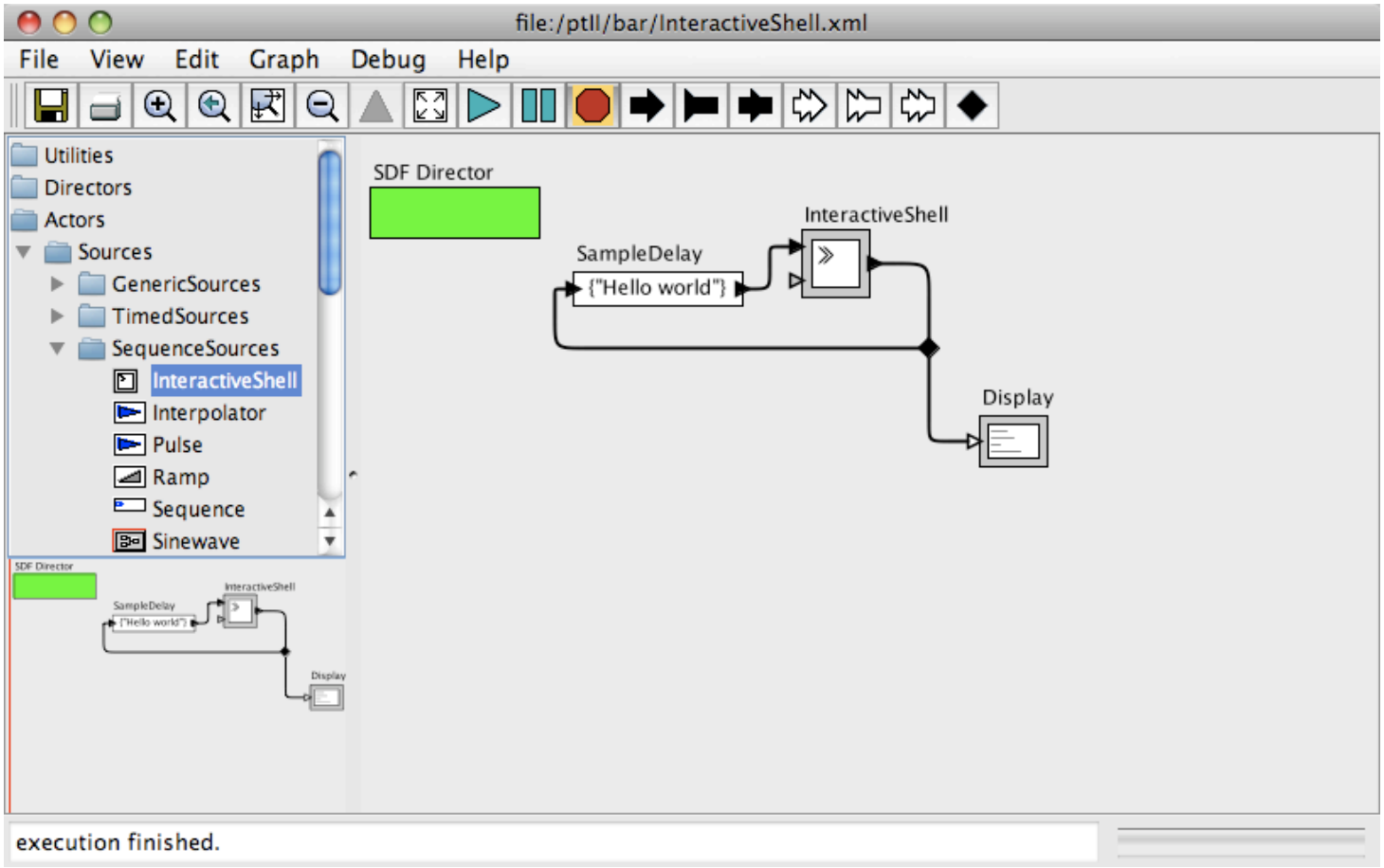
References

- Ptolemy project home page:
  http://ptolemy.eecs.berkeley.edu/
- Tutorial: Building Ptolemy II Models Graphically:
  http://www.eecs.berkeley.edu/Pubs/TechRpts/2007/EECS-2007-129.html
- Latest release:
  http://ptolemy.eecs.berkeley.edu/ptolemyII/ptIIlatest/
- Latest version in the SVN repository:
  http://chess.eecs.berkeley.edu/ptexternal/

Outline

- Simple model building
- Writing actors
- Writing directors

Building Models



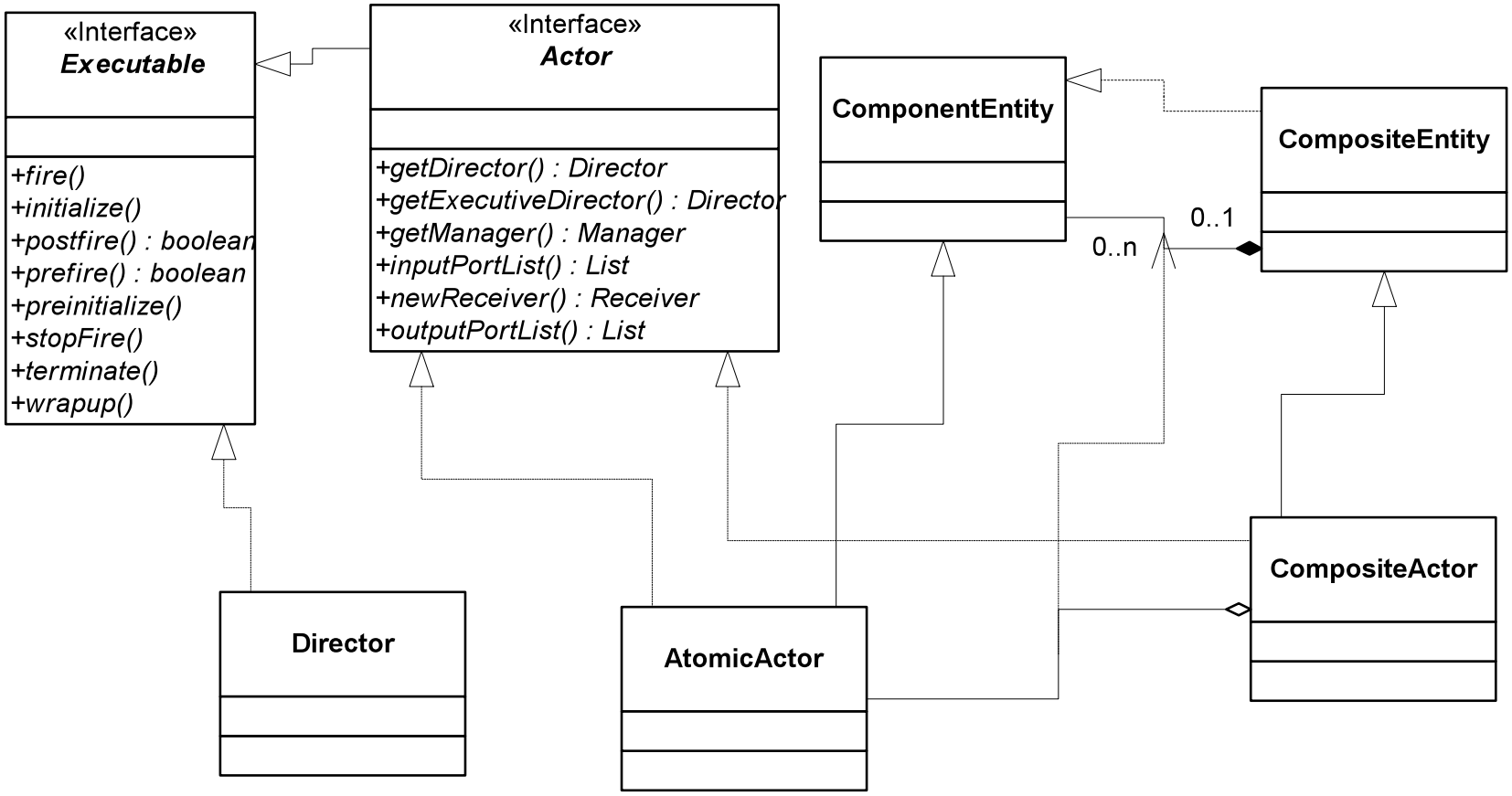
Outline

- Simple model building
- Writing actors
- Writing directors

Simple String Manipulation Actor

```
public class Ptolemnizer extends TypedAtomicActor {
    public Ptolemnizer(CompositeEntity container, String name)
            throws IllegalActionException, NameDuplicationException {
        super(container, name);
        input = new TypedIOPort(this, "input");
        input.setTypeEquals(BaseType.STRING);
        input.setInput(true);
        output = new TypedIOPort(this, "output");
        output.setTypeEquals(BaseType.STRING);
        output.setOutput(true);
    }
    public TypedIOPort input;
    public TypedIOPort output;
    public void fire() throws IllegalActionException {
        if (input.hasToken(0)) {
            Token token = input.get(0);
            String result = ((StringToken)token).stringValue();
            result = result.replaceAll("t", "pt");
            output.send(0, new StringToken(result));
        }
    }
}
```

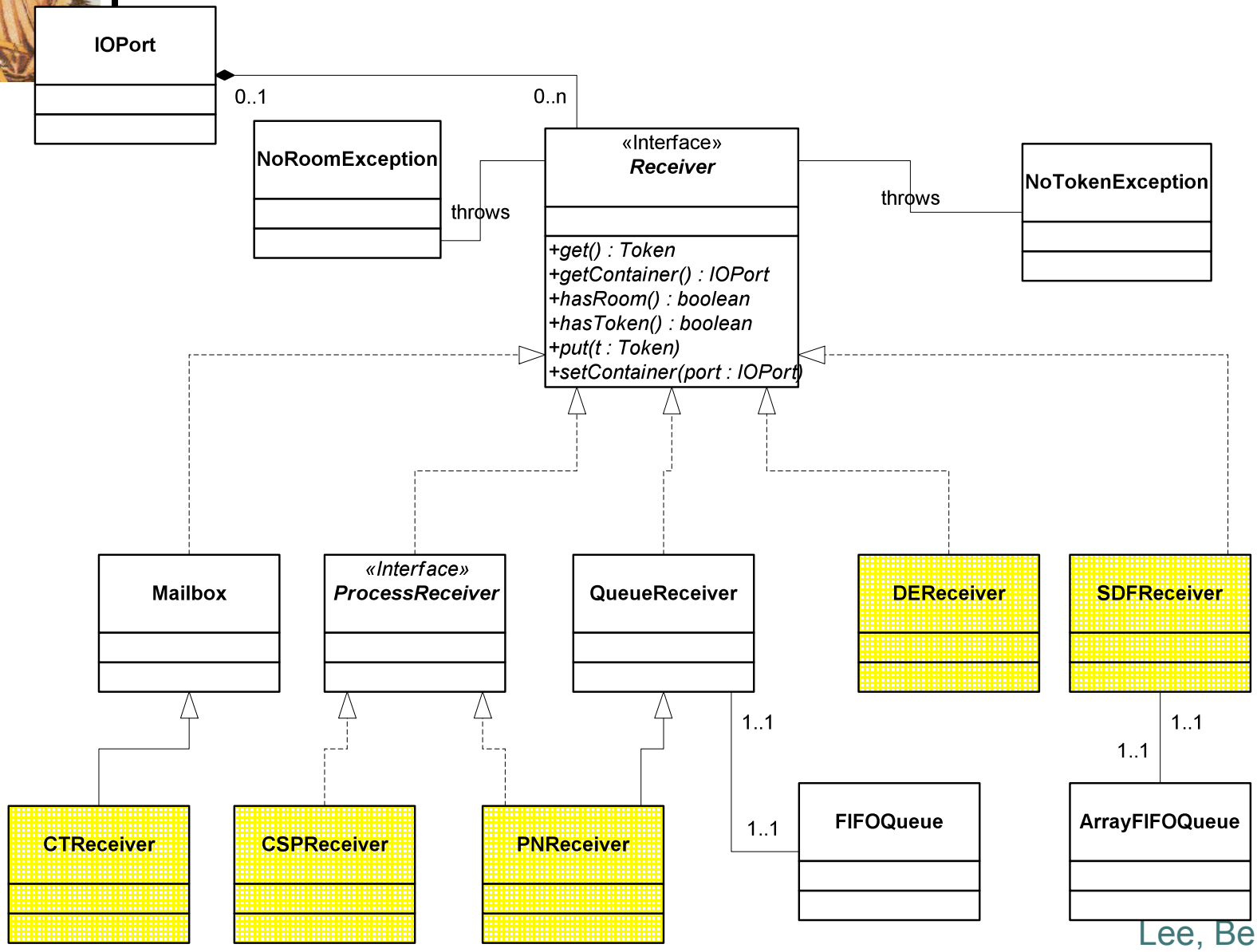
Object Model for Executable Components



Outline

- Simple model building
- Writing actors
- Writing directors

Object Model (Simplified) for Communication Infrastructure

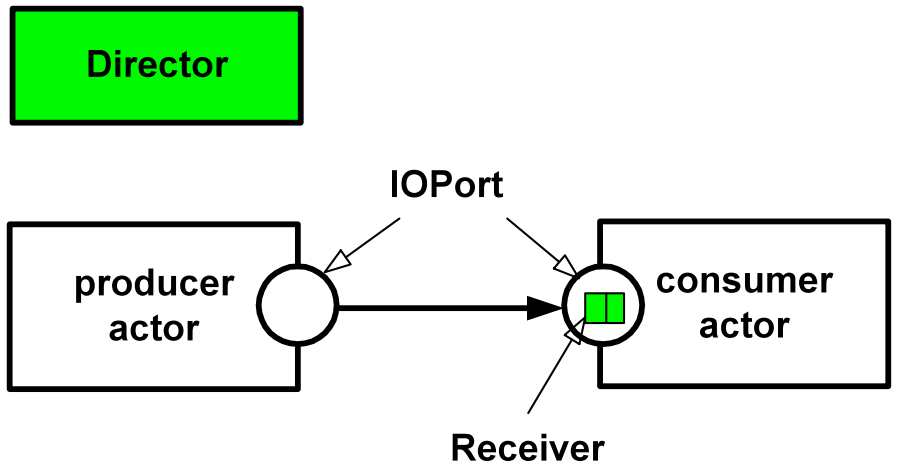


Object-Oriented Approach to Achieving Behavioral Polymorphism

«Interface»
Receiver

+get() : Token
+getContainer() : IOPort
+hasRoom() : boolean
+hasToken() : boolean
+put(t : Token)
+setContainer(port : IOPort)

These polymorphic methods implement the communication semantics of a domain in Ptolemy II. The receiver instance used in communication is supplied by the director, not by the component.



Recall: Behavioral polymorphism is the idea that components can be defined to operate with multiple models of computation and multiple middleware frameworks.

Extension Exercise

Build a director that subclasses PNDirector to allow ports to alter the "blocking read" behavior. In particular, if a port has a parameter named "tellTheTruth" then the receivers that your director creates should "tell the truth" when hasToken() is called. That is, instead of always returning true, they should return true only if there is a token in the receiver.

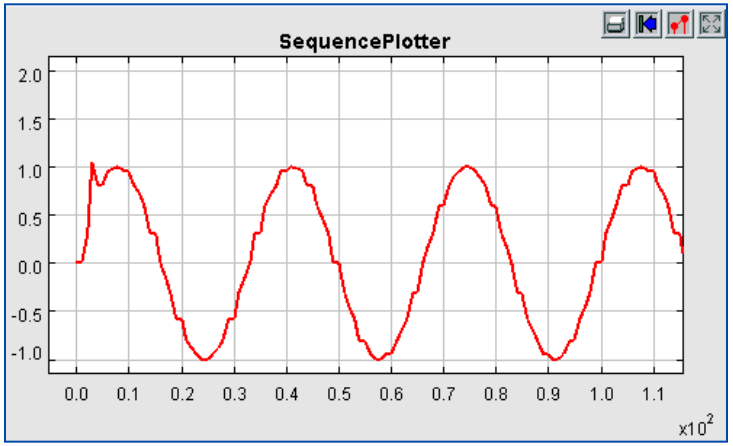
Parameterizing the behavior of a receiver is a simple form of communication refinement, a key principle in, for example, Metropolis.

Implementation of the NondogmaticPNDirector

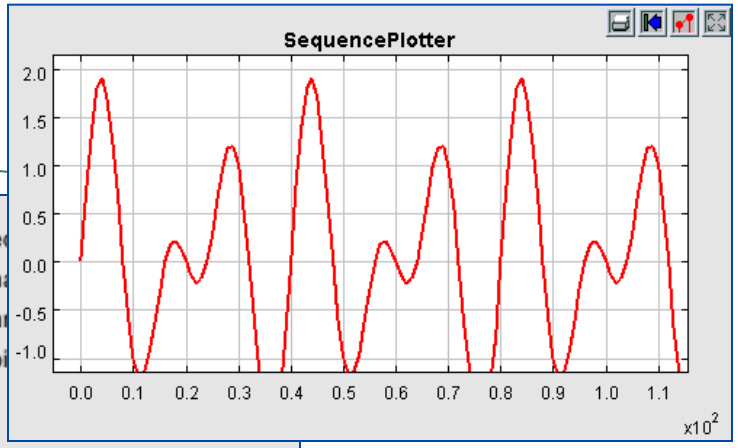
```
package doc.tutorial;
import …
public class NondogmaticPNDirector extends PNDirector {
    public NondogmaticPNDirector(CompositeEntity container, String name)
            throws IllegalActionException, NameDuplicationException {
        super(container, name);
    }
    public Receiver newReceiver() {
        return new FlexibleReceiver();
    }
    public class FlexibleReceiver extends PNQueueReceiver {
        public boolean hasToken() {
            IOPort port = getContainer();
            Attribute attribute = port.getAttribute("tellTheTruth");
            if (attribute == null) {
                return super.hasToken();
            }
            // Tell the truth...
            return _queue.size() > 0;
        }
    }
}
```

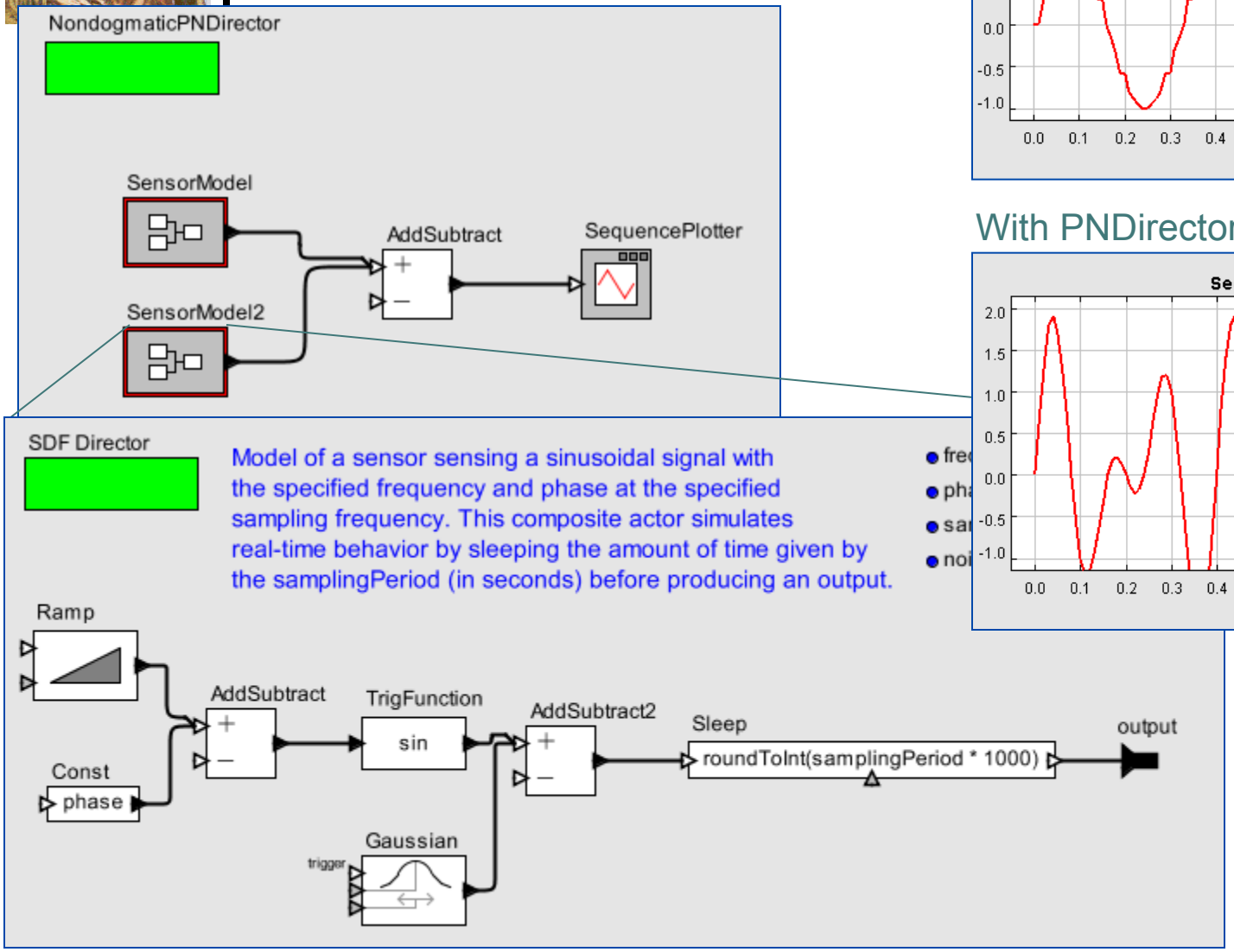
Using It

With NondogmaticPNDirector:



With PNDirector:





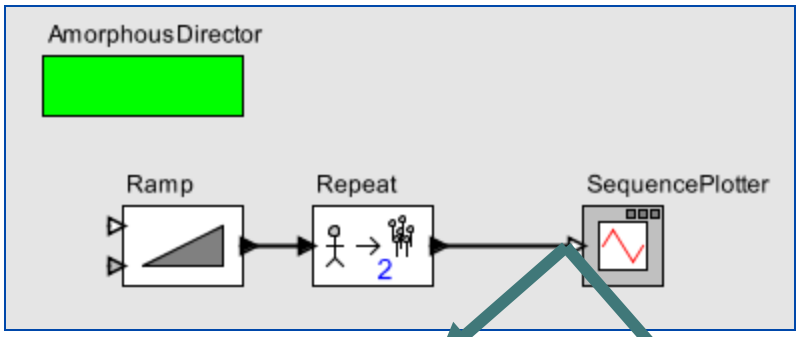
Extension Exercise 2

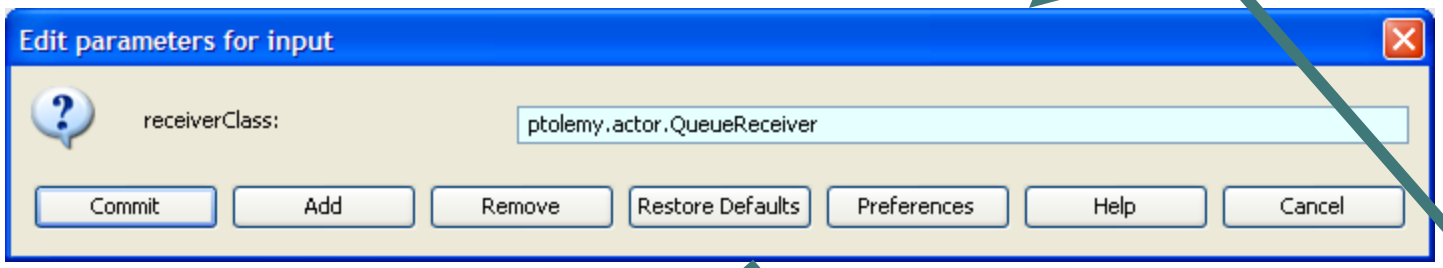
Build a director that subclasses Director and allows different receiver classes to be used on different connections. This is a form of what we call “amorphous heterogeneity.”

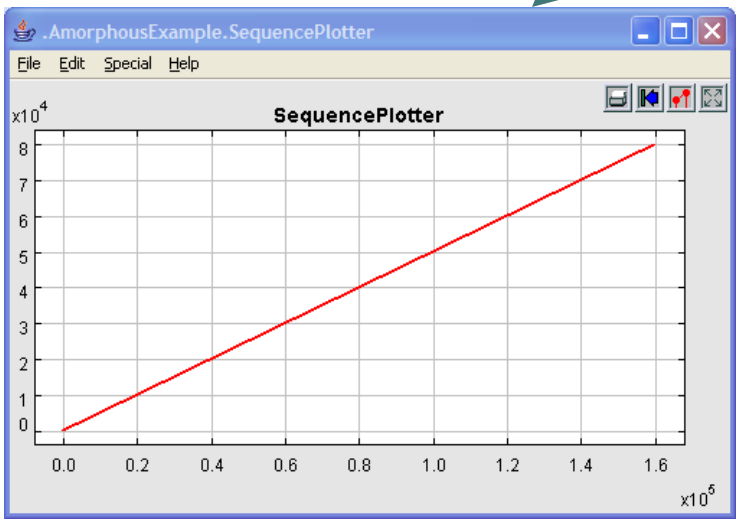
Implementation of the AmorphousDirector

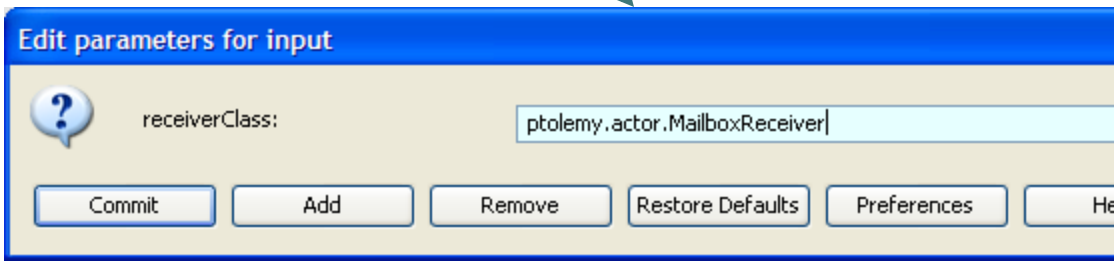
```
package doc.tutorial;
import …
public class AmorphousDirector extends Director {
    public AmorphousDirector(CompositeEntity container, String name)
            throws IllegalActionException, NameDuplicationException {
        super(container, name);
    }
    public Receiver newReceiver() {
        return new DelegatingReceiver();
    }
    public class DelegatingReceiver extends AbstractReceiver {
        private Receiver _receiver;
        public DelegatingReceiver() {
            super();
            _receiver = new SDFReceiver();
        }
        public DelegatingReceiver(IOPort container) throws IllegalActionException {
            super(container);
            _receiver = new SDFReceiver(container);
        }

        public void clear() throws IllegalActionException {
            IOPort container = getContainer();
            if (container != null) {
                StringParameter receiverClass = (StringParameter)
                        container.getAttribute("receiverClass", StringParameter.class);
                if (receiverClass != null) {
                    String className = ((StringToken)receiverClass.getToken()).stringValue();
                    try {
                        Class desiredClass = Class.forName(className);
                        _receiver = (Receiver)desiredClass.newInstance();
                    } catch (Exception e) {
                        throw new IllegalActionException(container, e,
                        "Invalid class for receiver: " + className);
                    }
                }
            }
            _receiver.clear();
        }

        public Token get() throws NoTokenException {
            return _receiver.get();
        }
        …
```

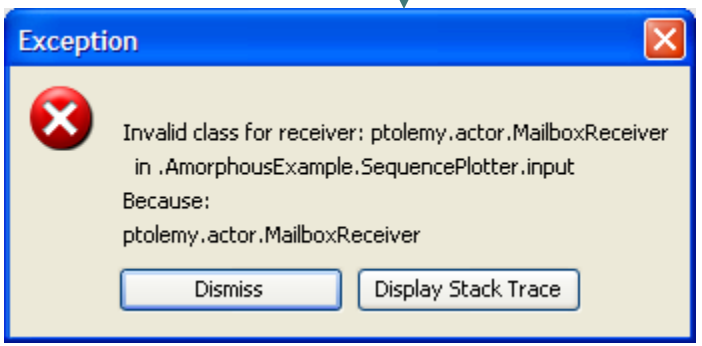
Using It


AmorphousDirector
Ramp
Repeat
2
SequencePlotter


Edit parameters for input
receiverClass:
ptolemy.actor.QueueReceiver
Commit
Add
Remove
Restore Defaults
Preferences
Help
Cancel


.AmorphousExample.SequencePlotter
File Edit Special Help
SequencePlotter
x10^4
8
7
6
5
4
3
2
1
0
0.0 0.2 0.4 0.6 0.8 1.0 1.2 1.4 1.6
x10^5


Edit parameters for input
receiverClass:
ptolemy.actor.MailboxReceiver
Commit
Add
Remove
Restore Defaults
Preferences


Exception
Invalid class for receiver: ptolemy.actor.MailboxReceiver
in .AmorphousExample.SequencePlotter.input
Because:
ptolemy.actor.MailboxReceiver
Dismiss
Display Stack Trace

Extension Exercise 3

Build a director that fires actors in left-to-right order, as they are laid out on the screen.

```
package doc.tutorial;
import java.util.Comparator;
import …
public class LeftRightDirector extends StaticSchedulingDirector {
    public LeftRightDirector(CompositeEntity container, String name) … {
        super(container, name);
        setScheduler(new LeftRightScheduler(this, "LeftRightScheduler"));
    }
    public class LeftRightScheduler extends Scheduler {
        public LeftRightScheduler(LeftRightDirector director, String name) … {
            super(director, name);
        }
        protected Schedule _getSchedule() … {
            StaticSchedulingDirector director = (StaticSchedulingDirector) getContainer();
            CompositeActor compositeActor = (CompositeActor) (director.getContainer());
            List actors = compositeActor.deepEntityList();
            Iterator actorIterator = actors.iterator();
            TreeSet sortedActors = new TreeSet(new LeftRightComparator());
            while (actorIterator.hasNext()) {
                Actor actor = (Actor) actorIterator.next();
                sortedActors.add(actor);
            }
            Schedule schedule = new Schedule();
            Iterator sortedActorsIterator = sortedActors.iterator();
            while (sortedActorsIterator.hasNext()) {
                Actor actor = (Actor) sortedActorsIterator.next();
                Firing firing = new Firing();
                firing.setActor(actor);
                schedule.add(firing);
            }

            return schedule;
        }
        public class LeftRightComparator implements Comparator {

            public int compare(Object o1, Object o2) {
                ...
            }
            public boolean equals(Object o) {
                …
            }
        }
    }
}
```

Implementation of the LeftRightDirector

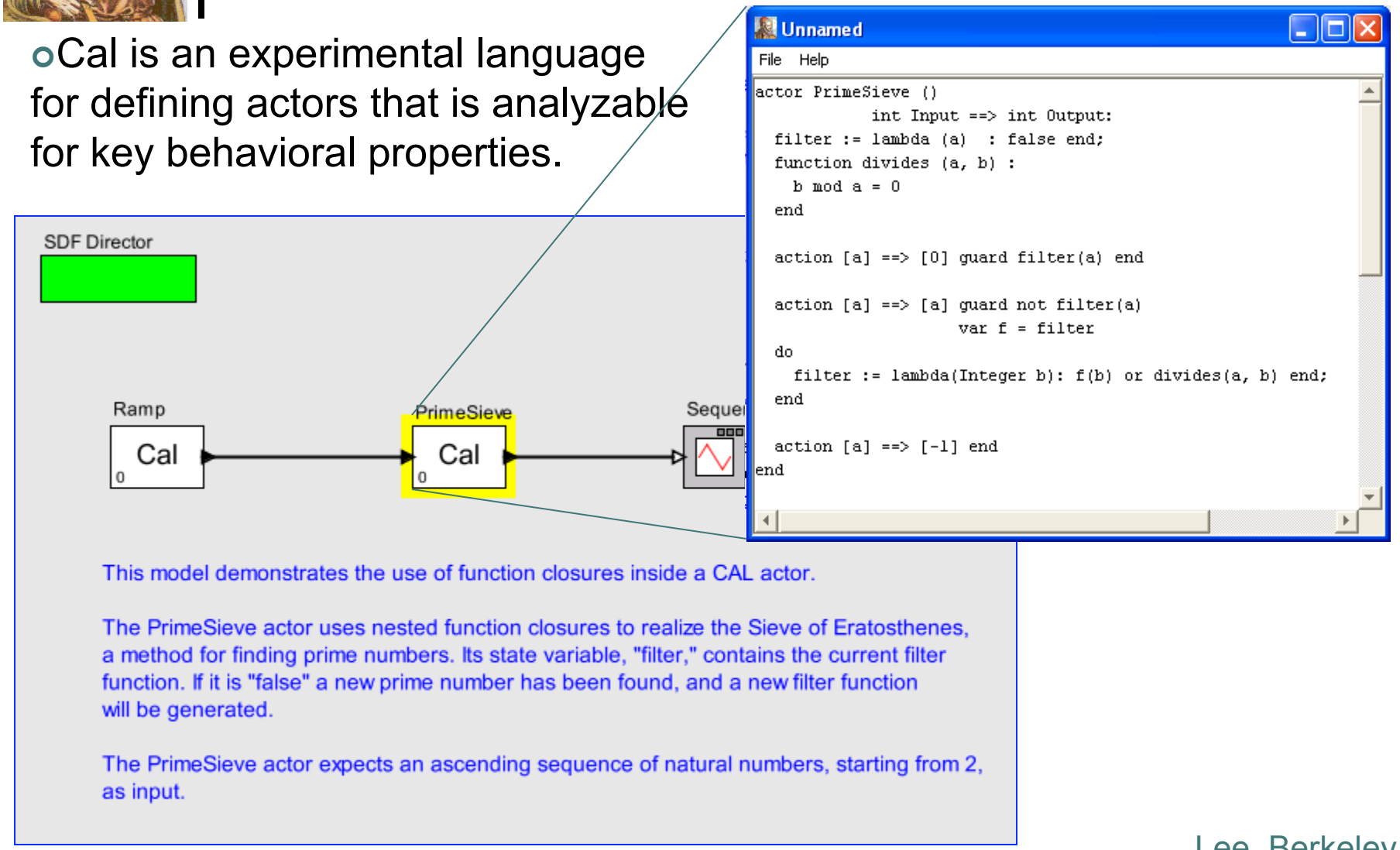
Ptolemy II Extension Points

- 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

- All of our “domains” are extensions built on a core infrastructure.

Example Extensions
Python Actors, Cal Actors, MATLAB Actors

- Cal is an experimental language for defining actors that is analyzable for key behavioral properties.

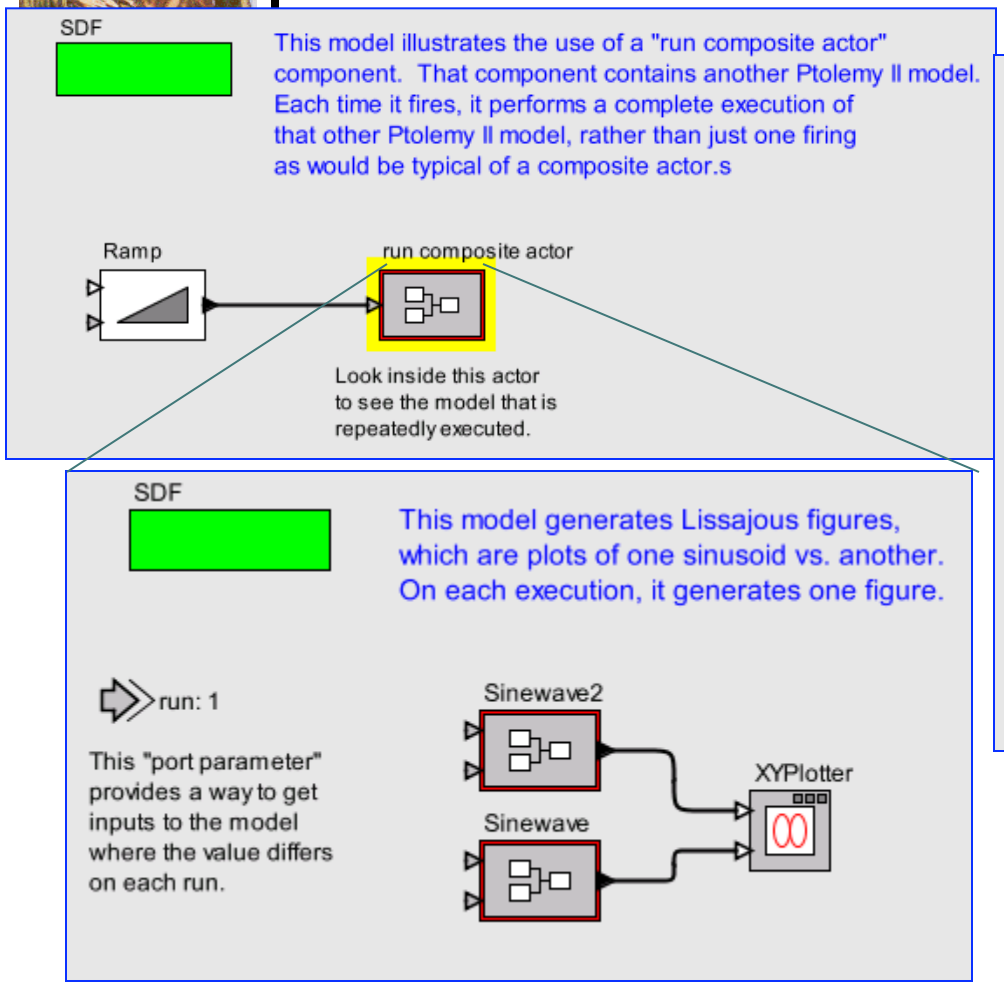


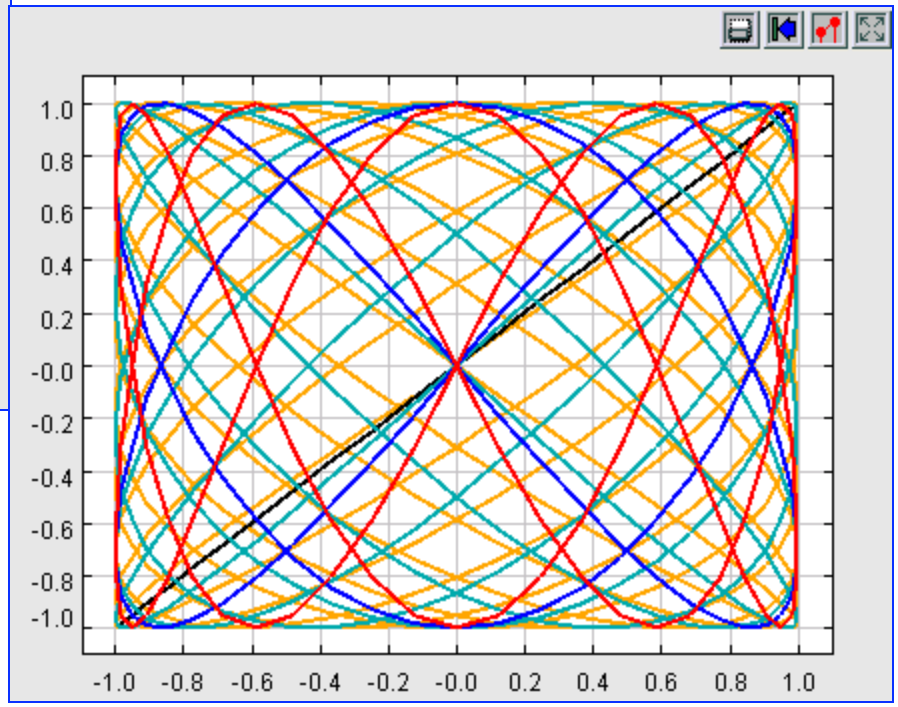
Ptolemy II Extension Points

- 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

- All of our “domains” are extensions built on a core infrastructure.

Example Extensions
Using Models to Control Models

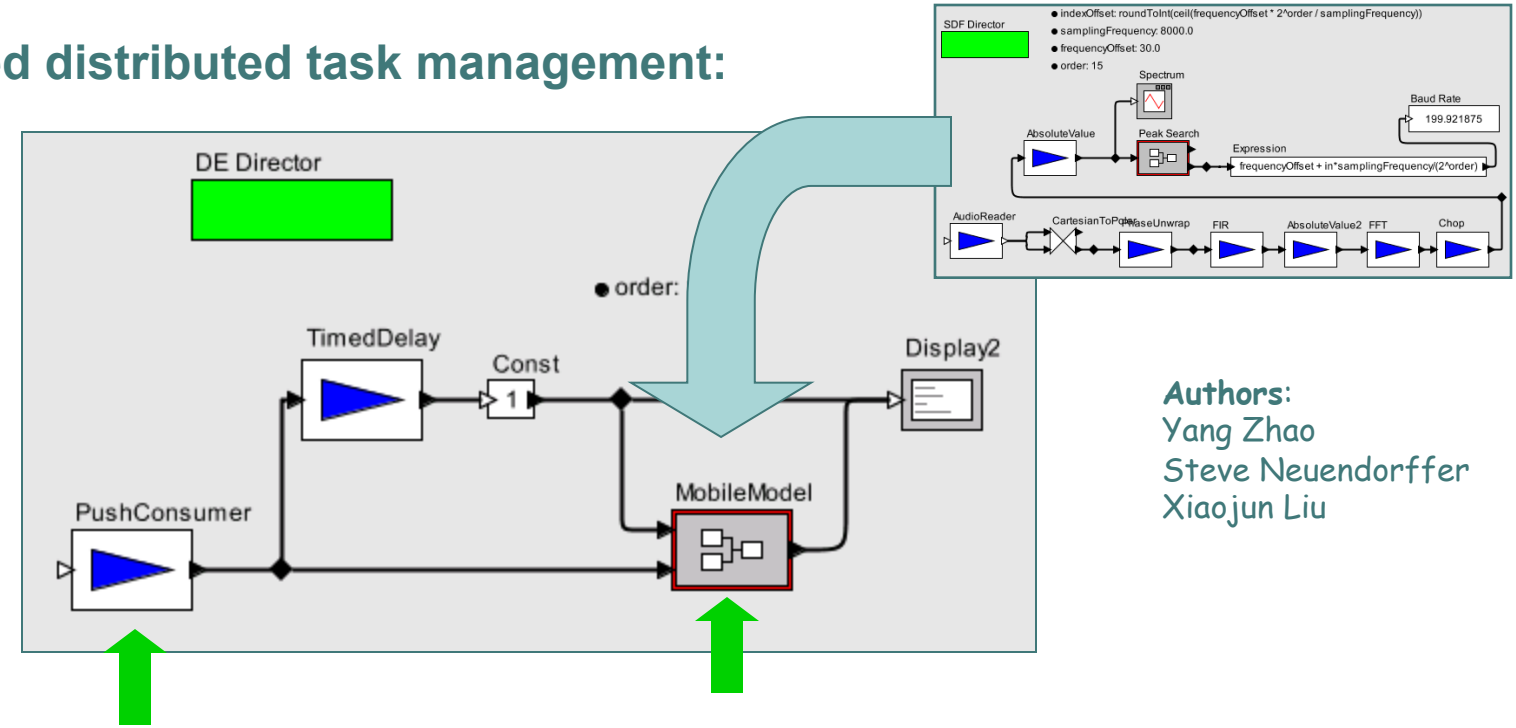




- This is an example of a “higher-order component,” or an actor that references one or more other actors.

Examples of Extensions
Mobile Models

Model-based distributed task management:



Authors:
Yang Zhao
Steve Neuendorffer
Xiaojun Liu

PushConsumer actor receives pushed data provided via CORBA, where the data is an XML model of a signal analysis algorithm.

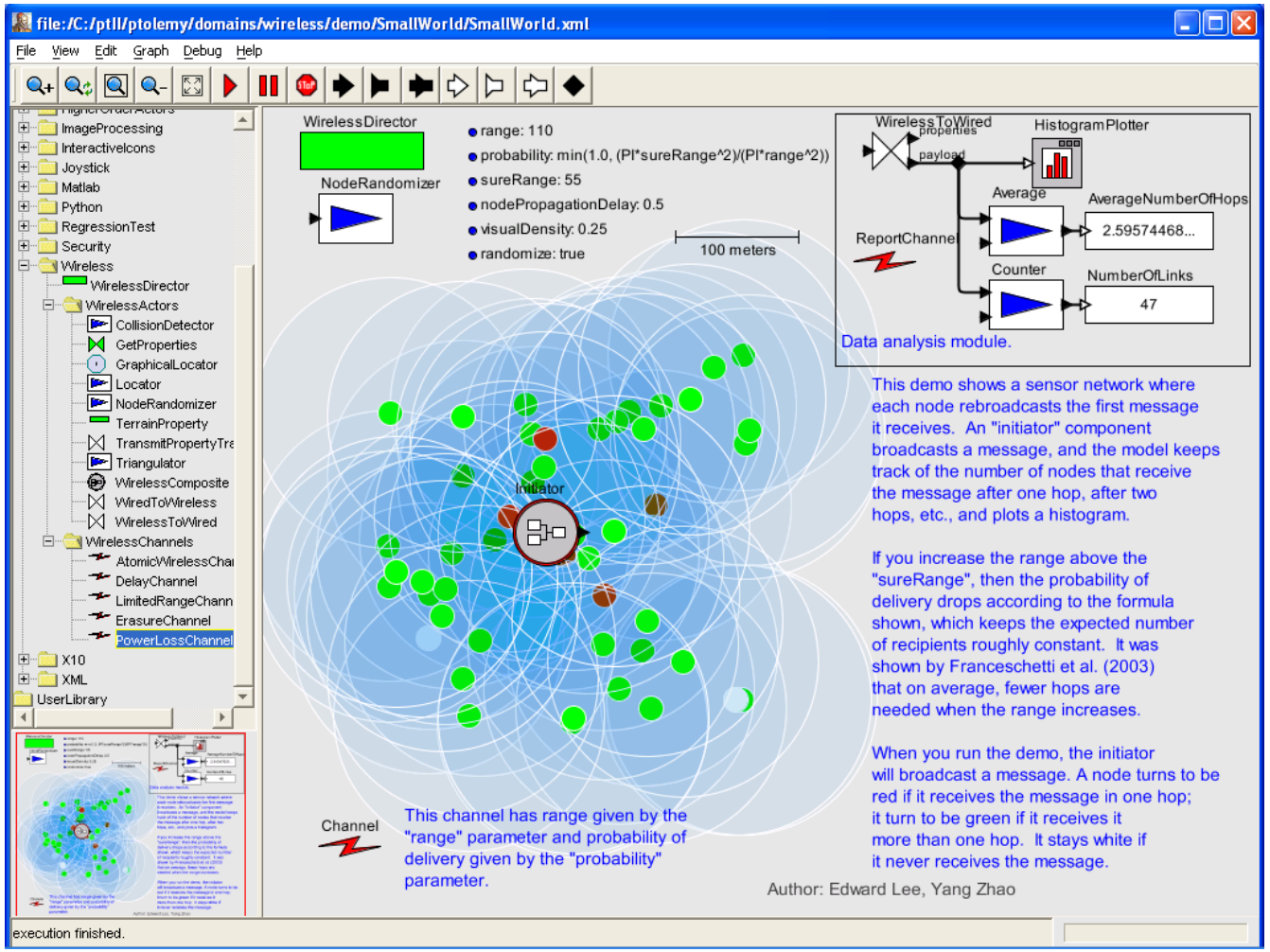
MobileModel actor accepts a StringToken containing an XML description of a model. It then executes that model on a stream of input data.

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

All of our “domains” are extensions built on a core infrastructure.

Extension of Discrete-Event Modeling for Wireless Sensor Nets

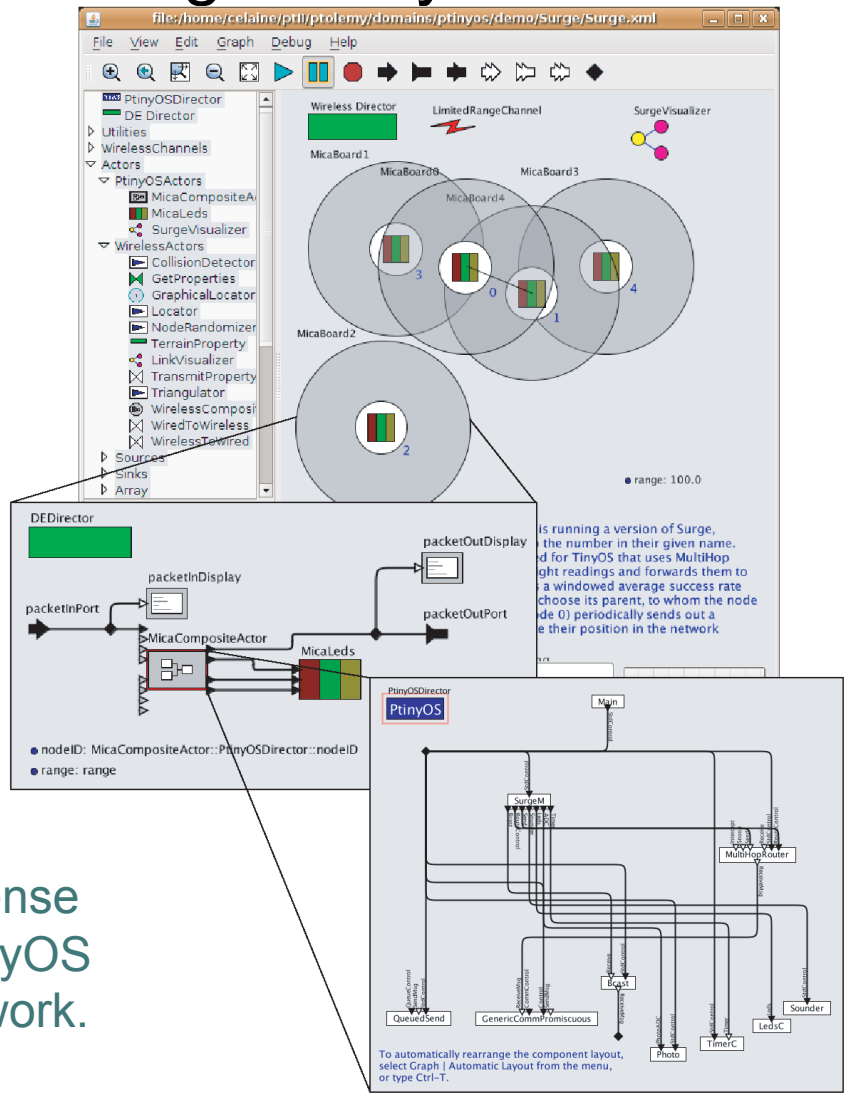


VisualSense extends the Ptolemy II discrete-event domain with communication between actors representing sensor nodes being mediated by a *channel*, which is another actor.

The example at the left shows a grid of nodes that relay messages from an *initiator* (center) via a channel that models a low (but non-zero) probability of long range links being viable.

Viptos: Extension of VisualSense with Programming of TinyOS nodes

Viptos demo:
Multihop routing (Surge)



Physical environment

Simulation
(with visualization of routing tree)

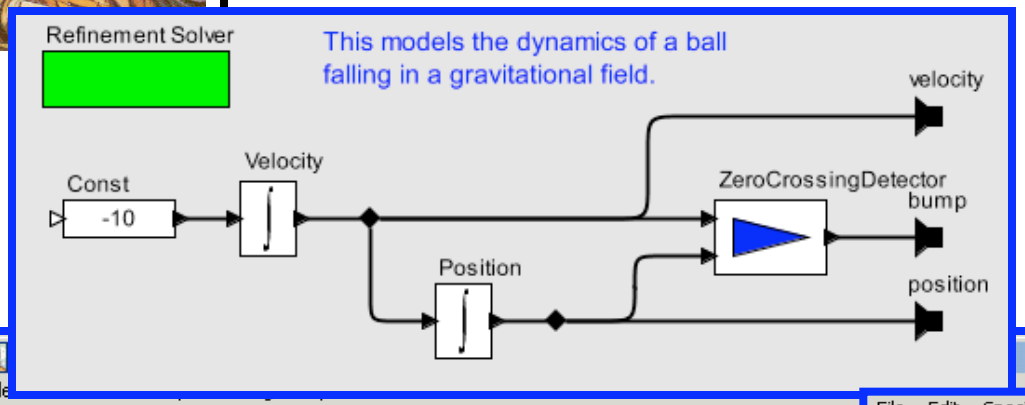
Hardware

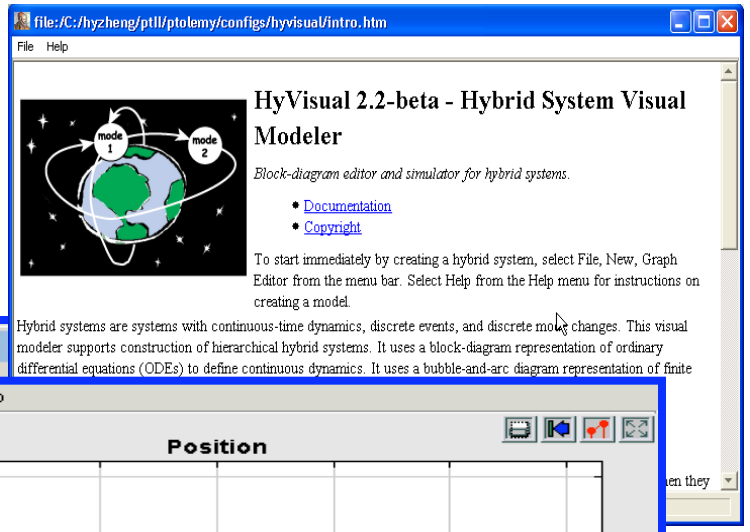
Software

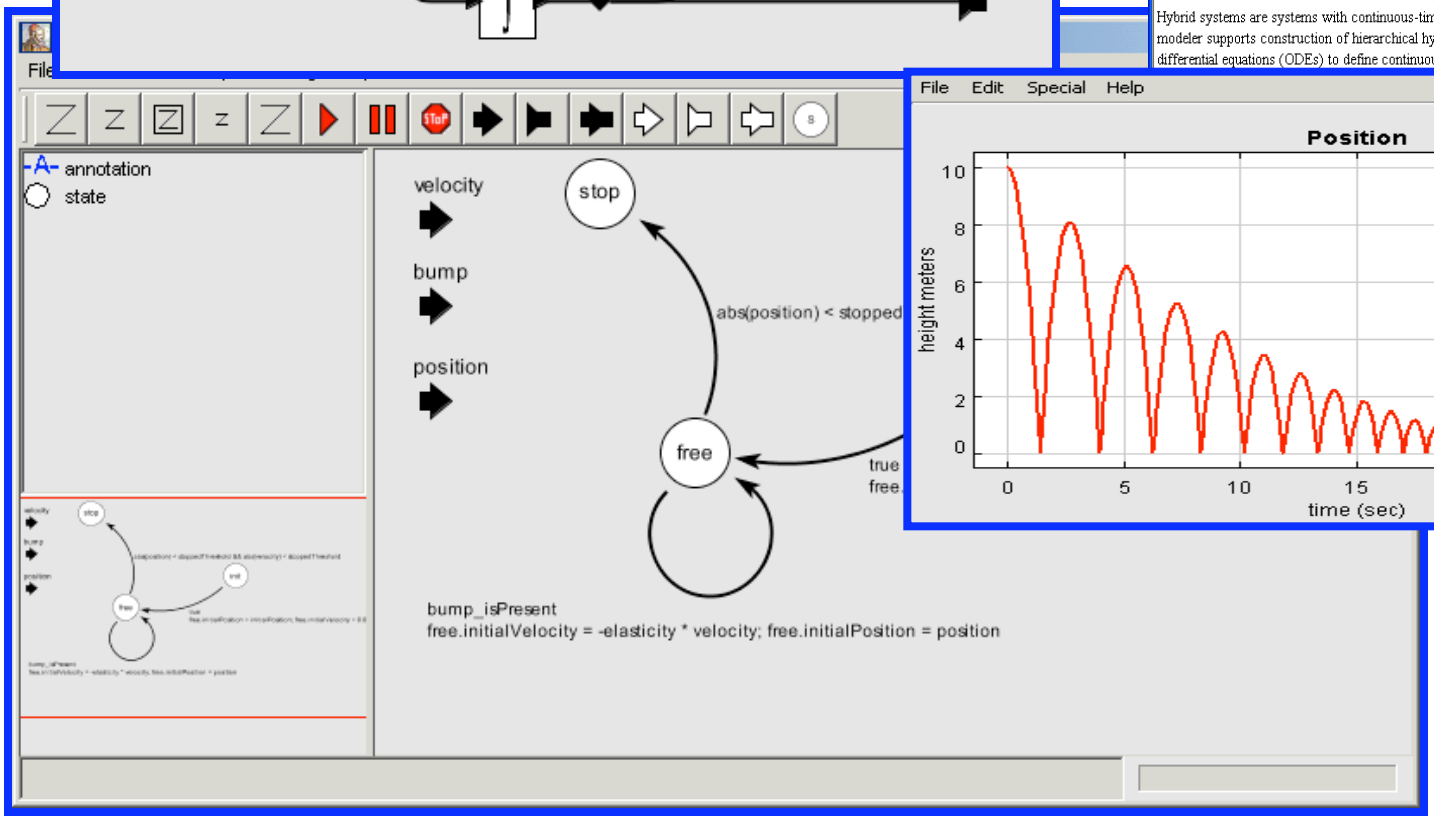
Code generation:
Models to nesC.

Viptos extends VisualSense with programming of TinyOS nodes in a wireless network. See the Ph.D. thesis of Elaine Cheong (Aug 2007).

Another Extension: HyVisual – Hybrid System Modeling Tool Based on Ptolemy II





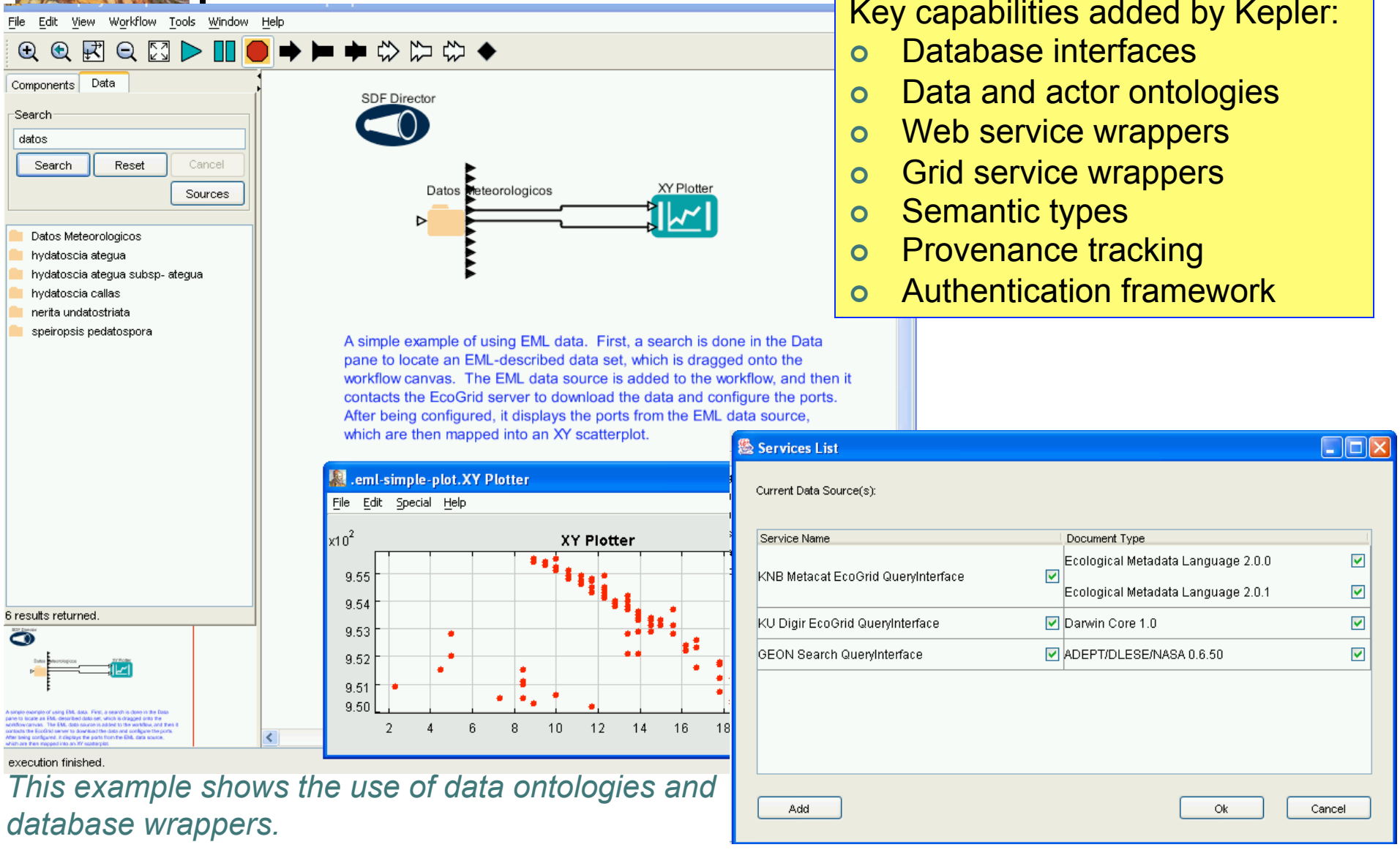


HyVisual was first released in January 2003.

Another Extension: Kepler: Aimed at Scientific Workflows

Key capabilities added by Kepler:

- Database interfaces
- Data and actor ontologies
- Web service wrappers
- Grid service wrappers
- Semantic types
- Provenance tracking
- Authentication framework

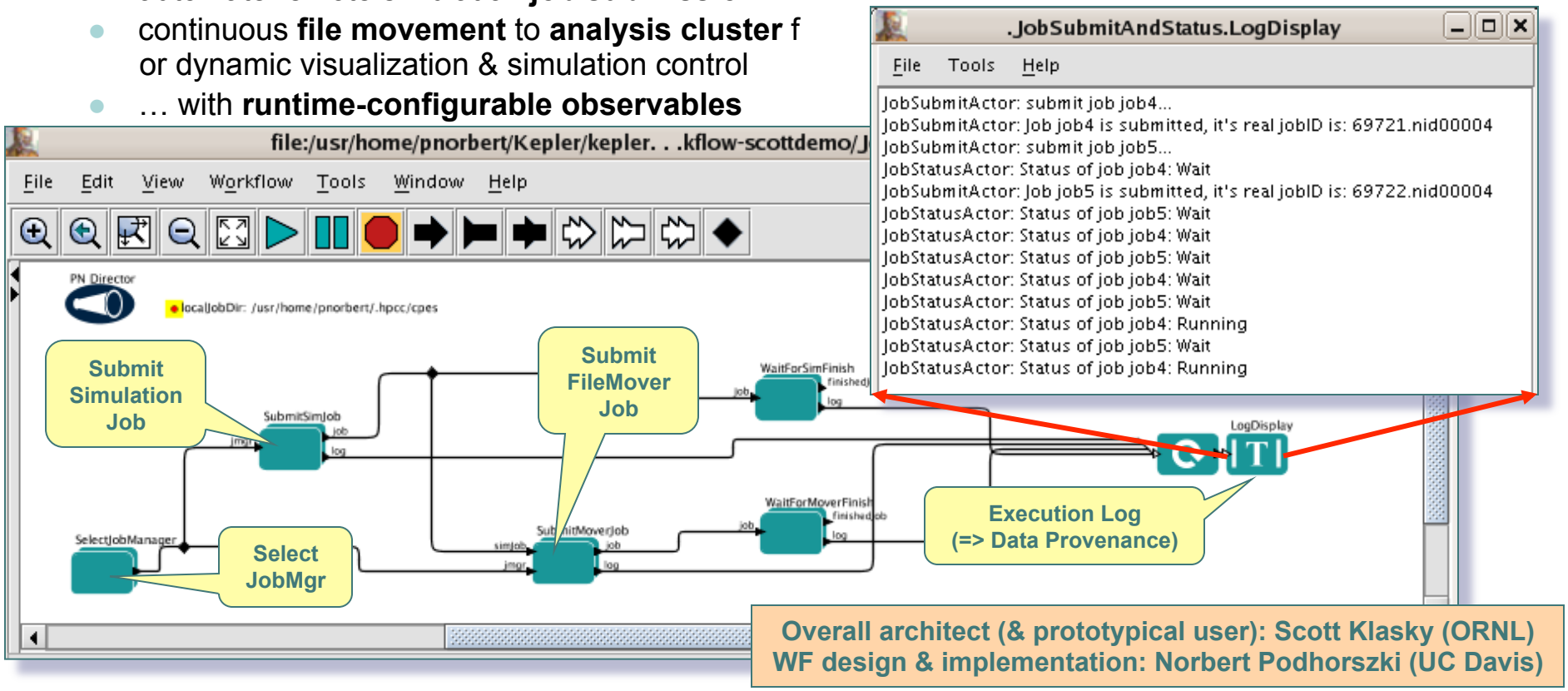


This example shows the use of data ontologies and database wrappers.

Kepler as an Interface to the Grid

CPES Fusion Simulation Workflow

- **Fusion Simulation Codes**: (a) GTC; (b) XGC with M3D
  - e.g. (a) currently 4,800 (soon: 9,600) nodes Cray XT3; 9.6TB RAM; 1.5TB simulation data/run
- **GOAL**:
  - automate remote simulation **job submission**
  - continuous **file movement** to **analysis cluster** f or dynamic visualization & simulation control
  - … with **runtime-configurable observables**



Overall architect (& prototypical user): Scott Klasky (ORNL)
WF design & implementation: Norbert Podhorszki (UC Davis)

Leverage: Kepler is a Team Effort

SciDAC
Scientific Discovery through Advanced Computing
SDM

Ptolemy II

SEEK

GEON
The Geosciences Network

ROADNet

Kepler

Griddles

SKIDL

SRB

EOL

Resurgence

Cipres

NLADR

LOOKING

Other contributors:
- **Chesire (UK Text Mining Center)**
- **DART (Great Barrier Reef, Australia)**
- **National Digital Archives + UCSD-TV (US)**
- **…**

Contributor names and funding info are at the Kepler website: http://kepler-project.org

Getting More Information: Documentation

PTOLEMY II

HETEROGENEOUS CONCURRENT MODELING AND DESIGN IN JAVA

Edited by:
Christopher Hylands, Edward A. Lee, Jie Liu, Xiaojun Liu, Steve Neuendorffer, Yuhong Xiong, Haiyang Zheng

VOLUME 1: INTRODUCTION TO PTOLEMY II

Authors:
Shuvra S. Bhattacharyya
Elaine Cheong
John Davis, II
Mudit Goel
Bart Kienhuis
Christopher Hylands
Edward A. Lee
Jie Liu
Xiaojun Liu
Lukito Muliadi
Steve Neuendorffer
John Reekie
Neil Smyth
Jeff Tsay
Brian Vogel
Winthrop Williams
Yuhong Xiong
Yang Zhao
Haiyang Zheng

Department of Electrical Engineering and Computer Sciences
University of California at Berkeley
http://ptolemy.eecs.berkeley.edu

Document Version 3.0
for use with Ptolemy II 3.0
June 8, 2003

Memorandum UCB/ERL M03/TBA
Earlier versions:
- *UCB/ERL M02/23*
- *UCB/ERL M99/40*
- *UCB/ERL M01/12*

This project is supported by the Defense Advanced Research Projects Agency (DARPA), the National Science Foundation, Chess (the Center for Hybrid and Embedded Software Systems), the State of California MICRO program, and the following companies: Agilent, Atmel, Cadence, Hitachi, Honeywell, National Semiconductor, Philips, and Wind River Systems..

Volume 1:

User-Oriented

PTOLEMY II

HETEROGENEOUS CONCURRENT MODELING AND DESIGN IN JAVA

Edited by:
Christopher Hylands, Edward A. Lee, Jie Liu, Xiaojun Liu, Steve Neuendorffer, Yuhong Xiong, Haiyang Zheng

VOLUME 2: PTOLEMY II SOFTWARE ARCHITECTURE

Authors:
Shuvra S. Bhattacharyya
Elaine Cheong
John Davis, II
Mudit Goel
Bart Kienhuis
Christopher Hylands
Edward A. Lee
Jie Liu
Xiaojun Liu
Lukito Muliadi
Steve Neuendorffer
John Reekie
Neil Smyth
Jeff Tsay
Brian Vogel
Winthrop Williams
Yuhong Xiong
Yang Zhao
Haiyang Zheng

Department of Electrical Engineering and Computer Sciences
University of California at Berkeley
http://ptolemy.eecs.berkeley.edu

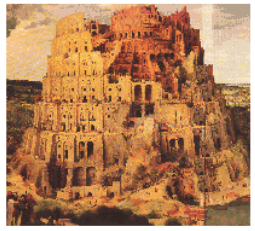
Document Version 3.0
for use with Ptolemy II 3.0
June 8, 2003

Memorandum UCB/ERL M03/TBA
Earlier versions:
- *UCB/ERL M02/23*
- *UCB/ERL M99/40*
- *UCB/ERL M01/12*

This project is supported by the Defense Advanced Research Projects Agency (DARPA), the National Science Foundation, Chess (the Center for Hybrid and Embedded Software Systems), the State of California MICRO program, and the following companies: Agilent, Atmel, Cadence, Hitachi, Honeywell, National Semiconductor, Philips, and Wind River Systems..

Volume 2:

Developer-Oriented



PTOLEMY II

HETEROGENEOUS CONCURRENT MODELING AND DESIGN IN JAVA

Edited by:
Christopher Hylands, Edward A. Lee, Jie Liu, Xiaojun Liu, Steve Neuendorffer, Yuhong Xiong, Haiyang Zheng

VOLUME 3: PTOLEMY II DOMAINS

Authors:
Shuvra S. Bhattacharyya
Elaine Cheong
John Davis, II
Mudit Goel
Bart Kienhuis
Christopher Hylands
Edward A. Lee
Jie Liu
Xiaojun Liu
Lukito Muliadi
Steve Neuendorffer
John Reekie
Neil Smyth
Jeff Tsay
Brian Vogel
Winthrop Williams
Yuhong Xiong
Yang Zhao
Haiyang Zheng

Department of Electrical Engineering and Computer Sciences
University of California at Berkeley
http://ptolemy.eecs.berkeley.edu

Document Version 3.0
for use with Ptolemy II 3.0
June 8, 2003

Memorandum UCB/ERL M03/TBA
Earlier versions:
- *UCB/ERL M02/23*
- *UCB/ERL M99/40*
- *UCB/ERL M01/12*

This project is supported by the Defense Advanced Research Projects Agency (DARPA), the National Science Foundation, Chess (the Center for Hybrid and Embedded Software Systems), the State of California MICRO program, and the following companies: Agilent, Atmel, Cadence, Hitachi, Honeywell, National Semiconductor, Philips, and Wind River Systems..

Volume 3:

Researcher-Oriented

Tutorial information: http://ptolemy/conferences/07/tutorial.htm