

Process-Based Software Components

Mobies Phase 1, UC Berkeley

Edward A. Lee and Tom Henzinger

(with contributions from Steve Neuendorffer, Christopher Hylands, Jie Liu, Xiaojun Liu, and Haiyang Zheng)

PI Meeting, Chandler, AZ

January 29, 2003

PI: Edward A. Lee, 510-642-0455, eal@eecs.berkeley.edu
Co-PI: Tom Henzinger, 510-643-2430, tah@eecs.berkeley.edu
PM: John Bay
Agent: James Lyttle, AFRL/IFSC, James.Lyttle@wpafb.af.mil
Award end date: December, 2003
Contract number: F33615-00-C-1703
AO #: J655

Mobies Phase 1 UC Berkeley 1

Subcontractors and Collaborators


- Subcontractor
 - Univ. of Maryland (C code generation)
- Collaborators
 - Caltech SEC (fan-driven platform)
 - UCB SEC (helicopters)
 - Kestrel (code generation technology)
 - Vanderbilt (HSIF)
 - Penn (HSIF)
 - CMU (HSIF)
 - Ford/GM/UCB (HSIF)
 - Research in Motion Limited
 - Brigham Young University (hardware generation)

Mobies Phase 1 UC Berkeley 2

Project Goals and Problem Description

Our focus is on component-based design using principled *models of computation* and their *runtime environments* for embedded systems. The emphasis of this project is on the dynamics of the components, including the communication protocols that they use to interface with other components, the modeling of their state, and their flow of control. The purpose of the mechanisms we develop is to improve robustness and safety while promoting component-based design.

Technical Approach Summary

- Models of computation
 - supporting heterogeneity
 - supporting real-time computation
 - codifications of design patterns
 - definition as *behavioral types*
- Co-compilation
 - joint compilation of components and architecture
 - vs. code generation
 - supporting heterogeneity
- Ptolemy II  **our tool**
 - our open-architecture software laboratory
 - shed light on models of computation & co-compilation
 - by prototyping modeling frameworks and techniques

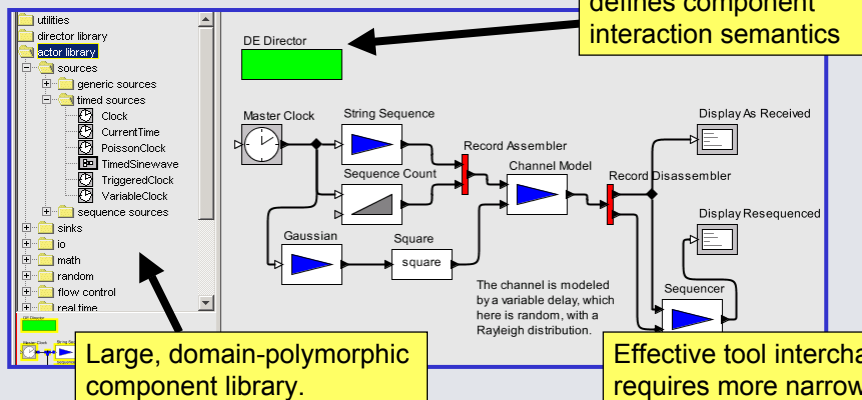
Summary of Accomplishments

- Heterogeneous modeling
 - HyVisual hybrid system modeler
 - Domain polymorphism concept & realization
 - Theory of responsible frameworks
 - Behavioral type system
 - Related Giotto, Simulink, and Timed Multitasking semantics
 - Component definition principles (Cal and Actif)
 - Component interaction (CI) domain
 - Higher-order expression language
- Tool integration
 - Charon import/export from Ptolemy II
 - HSIF import to HyVisual
 - Matlab integration with Ptolemy II
- Code generation
 - Co-compilation concept
 - Giotto program generation
 - Java code generation from SDF models
 - C code generation from Java
 - Code generation from expressions
 - FSM code generation

Mobies Phase 1 UC Berkeley 5

Tool Interchange - A Semantics Problem

Basic Ptolemy II infrastructure:



Mobies Phase 1 UC Berkeley 6

HyVisual - Hybrid System Modeling Tool Based on Ptolemy II

HyVisual Demo - Building a Hybrid System

Tool Customization Ptolemy II Configurations

- A Configuration is a Ptolemy II model, defined in an XML file, that contains:
 - A customized library of components
 - References to customized on-line docs
 - A list of factories for model editors. E.g.:
 - HSIF editor
 - MoML editor
 - HTML editor
 - Text editor
 - Template models for new designs. E.g.:
 - Blank model populated with a director
 - State refinement for modal models
 - Transition refinement for modal models

Mobies Phase 1 UC Berkeley 9

Reading HSIF files in HyVisual



HSIF file detected.

MoML file generated.

Invoking XSL Translation

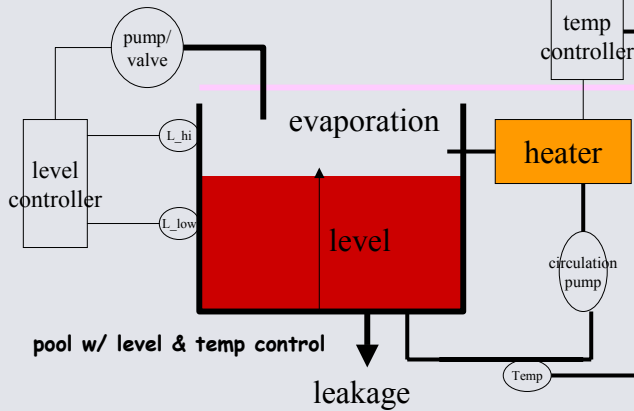
(HSIF → MoML of Ptolemy)

- Global variables → ports;
- Local variables → parameters; or ports controlled by dynamic equations;
- Hybrid Automata → modal models;
- Invariant expressions → invariant actors;
- Locations → states & refinements;
- Differential equations → refinement structures;
- Transitions → transitions.

Mobies Phase 1 UC Berkeley 10

Swimming Pool Example

Objective is to keep the water temperature and level of the swimming pool inside a proper range.



Parameters:

$f_{in}=0.2$ m/min
 $f_{leak}=0.04$ m/min
 (rates of change in level due to inflow and leak respectively)

$heat=1/60$ /min
 $ambient=1/120$ /min
 $inflow=1/100$ /min
 (rate of temperature change due to the heater, surroundings and incoming cold water)

$T_{max}=32$
 $T_{min}=27$
 $T_{heat}=40$
 $T_{inflow}=15$
 $T_{ambient}=25$
 (all in degrees centigrade)

$level_{max}=2.6$ m
 $level_{min}=2.2$ m
 pump on-off transition
 timeout = 0.5 min

A swimming pool model from Bruce Krogh.

Mobies Phase 1 UC Berkeley 11

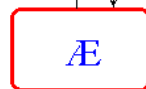
GME View

GME model constructed by Bruce Krogh.
 GME exports HSIF from this model.



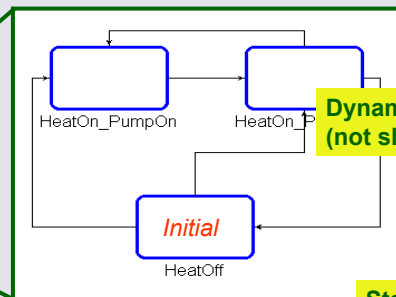
Thermostat

real level



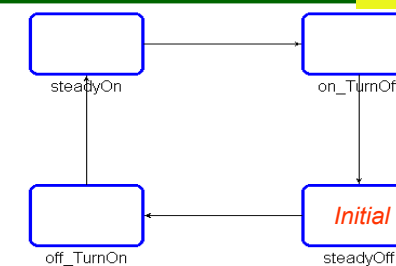
pump

Hybrid automata



Dynamics inside (not shown)

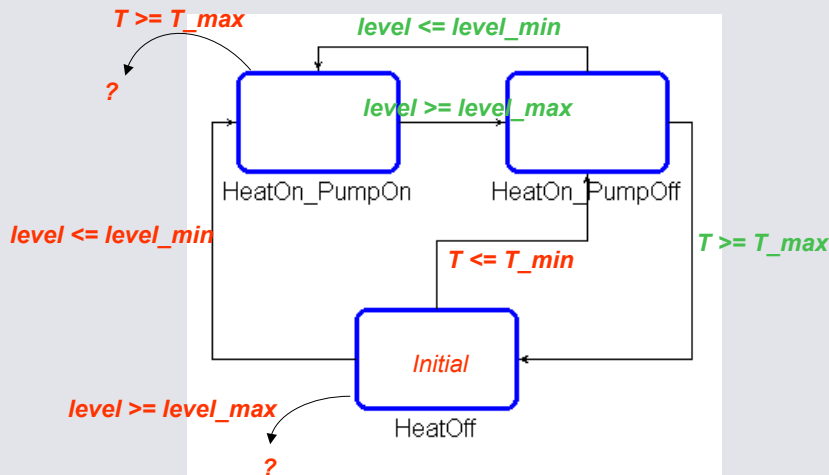
State machines



Berkeley 12

Original Thermostat State Machine

The detailed thermostat state machine model with non-deterministic behaviors.



Mobies Phase 1 UC Berkeley 13

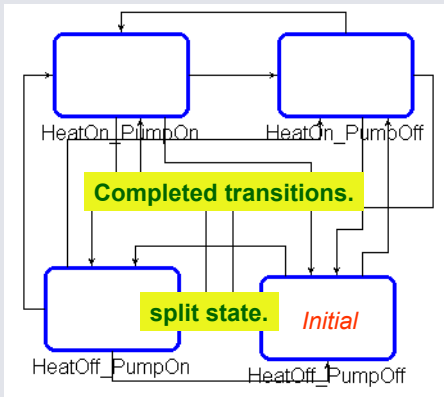
Issues Raised by This Model

- Model is nondeterministic
 - If temperature and level are both low, there are two enabled transitions from HeatOff
 - How/whether to simulate?
- HeatOn_PumpOff state is transient
 - If temperature and level are both low, then we can transition right through this state
 - Glitch in simulation?
- Initial conditions violated invariants
 - How to specify initial conditions for both simulation and reachability analysis?
 - How should a simulator deal with violated invariants?

Mobies Phase 1 UC Berkeley 14

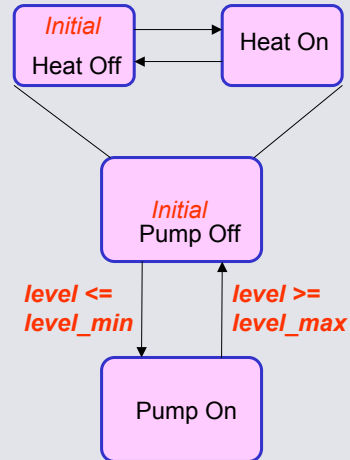
Modified Model

The refined thermostat model with deterministic behaviors.



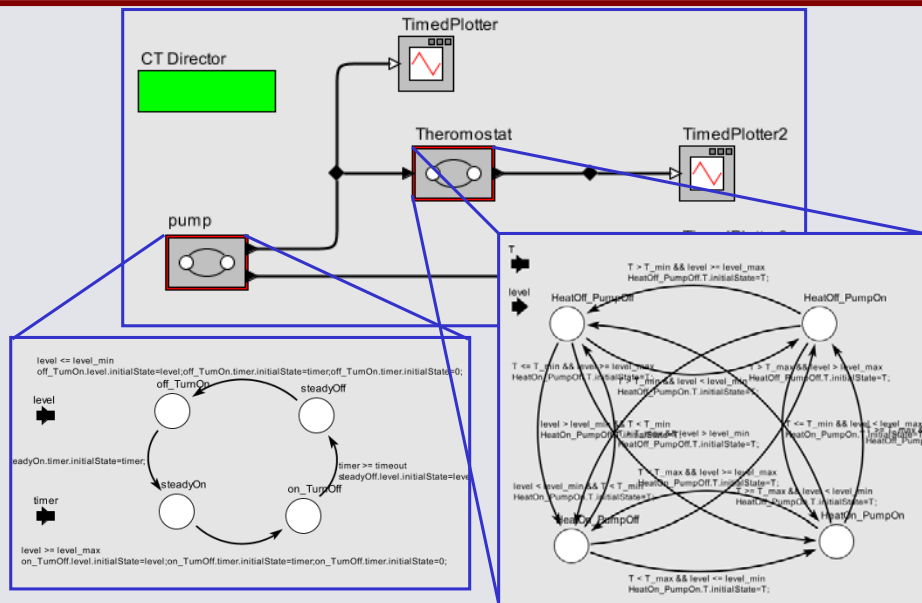
Made transitions mutually exclusive.

Hierarchical state machine not representable in HSIF.



Mobies Phase 1 UC Berkeley 15

Model After Import Into HyVisual



Manual Modifications Required

- Added display actors
- Visual layout of components
- Set simulation control parameters
 - Length of simulation time
 - Step size control
 - Choice of solver

More Issues Raised

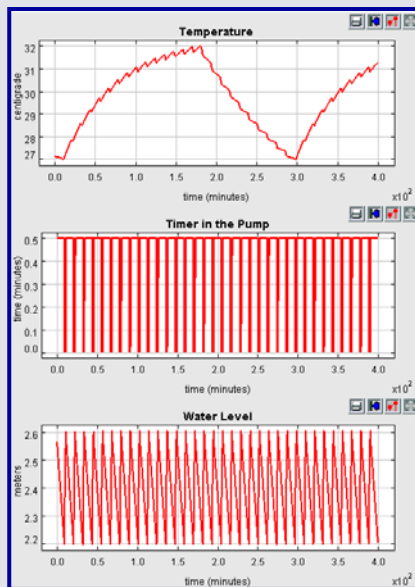
- Hierarchical state machines are not supported in HSIF
 - Better orthogonalization of designs
 - Need for transitions to optionally reset destination to the initial state.
- HSIF annotations needed?
 - One tool may need to put in annotations that will be and should be ignored by other tools. E.g.:
 - Simulation parameters
 - Identification of signals to display
 - Initial conditions or recorded state
- Should HSIF support transition refinements?

XSL Transformation Phases

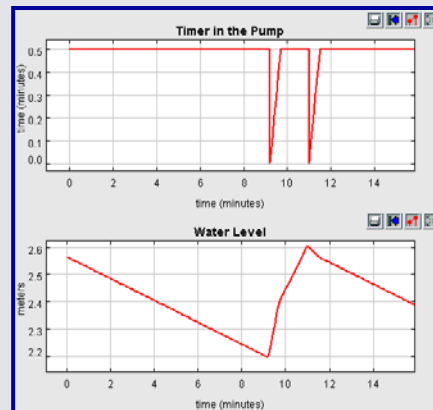
- Global variables:
 - Global variables become input or output ports
 - HSIF defines input, controlled, and observable
- Local variables:
 - Some are converted to parameters
 - If controlled by dynamic equations, converted to ports
- Mapping of components
 - HA to modal models
 - Transitions to transitions
 - Differential equations to state refinements
 - Invariants to invariant actors

Mobies Phase 1 UC Berkeley 19

Simulation Results



Timer used in Pump

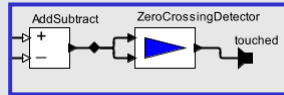


Water level

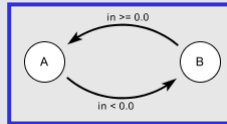
Mobies Phase 1 UC Berkeley 20

More Issues: Transition Semantics

- In continuous-time models, HyVisual uses *event detectors* to identify the precise time at which an event occurs:



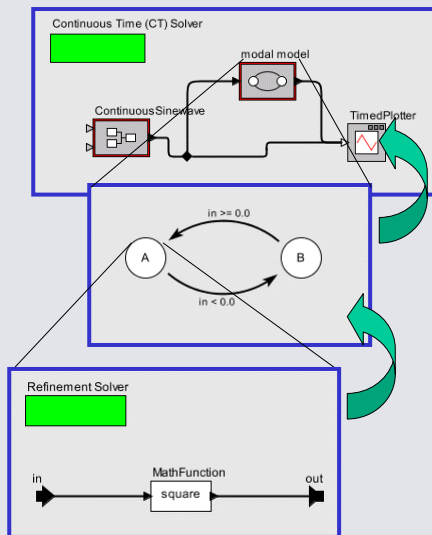
- Semantics of transitions, however, is to *enable* a mode change. Precise time of enablement is not identified:



- One consequence: deterministic model becomes nondeterministic if simulator takes steps that are too large.
- Another consequence: invariants may be violated due to failure to take mode transitions on time.

Mobies Phase 1 UC Berkeley 21

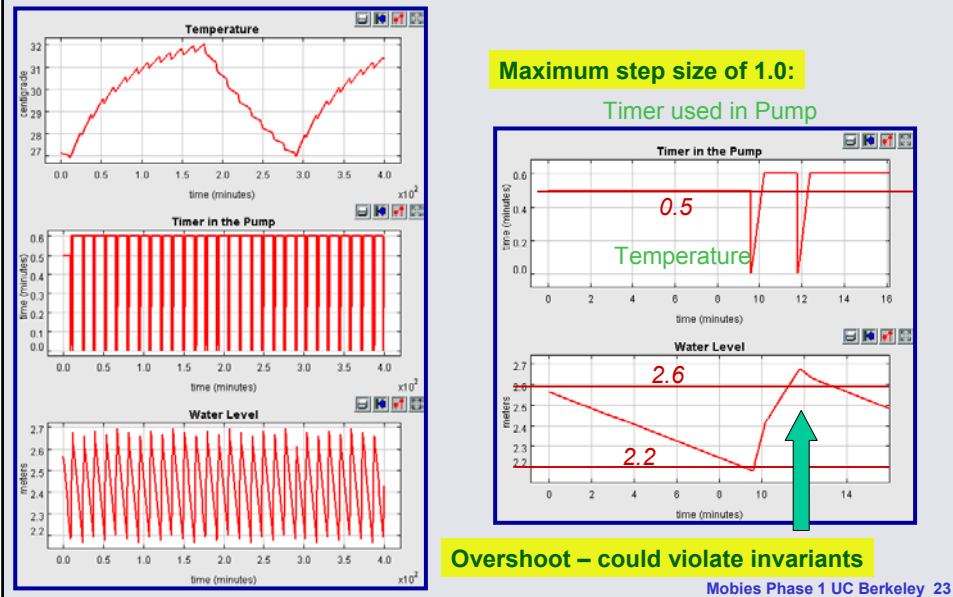
Model Errors vs. Exceptions



- Model errors are passed up the model hierarchy
- Exceptions are passed up the procedure-call stack
- Model errors can be used to refine step size while preserving information-hiding.

Mobies Phase 1 UC Berkeley 22

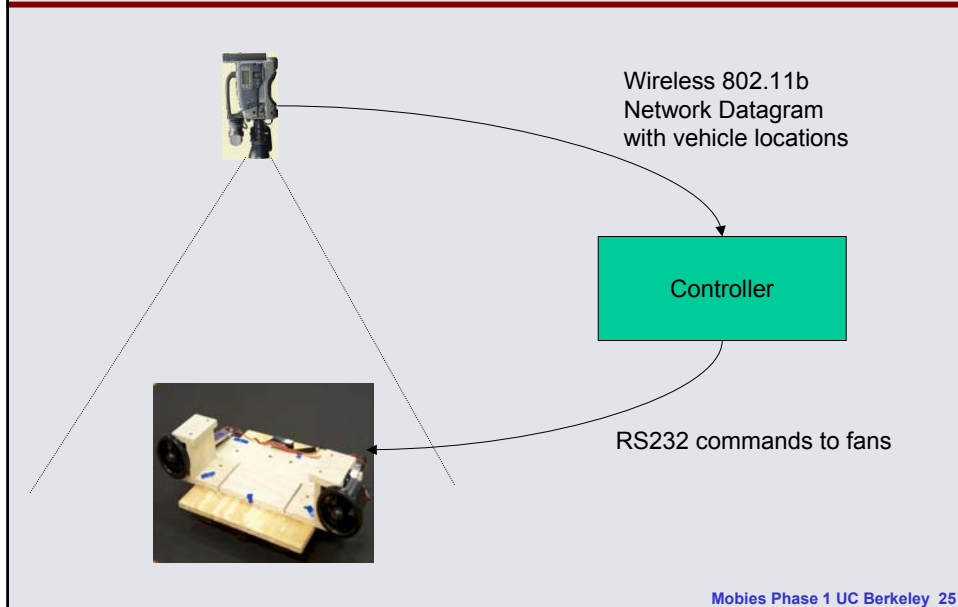
If Step-Size is Too Large...



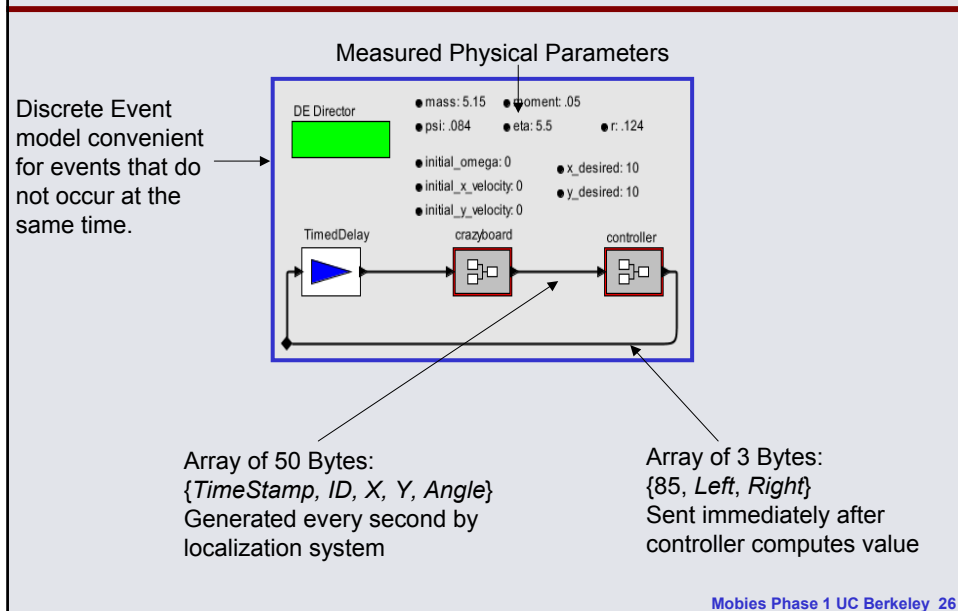
Code Generation Status Update

- Giotto code generator from Giotto domain
 - still need code generation from FSM to get modal models
- Java code generator from SDF domain
 - based on Soot compiler infrastructure (McGill)
 - type specialization
 - static scheduling, buffering
 - code substitution using model of computation semantics
- C code generation from Java
 - University of Maryland subcontract
 - based on Soot compiler infrastructure (McGill)
 - preliminary concept demonstration built
- Configurable hardware synthesis
 - targeted Wildcard as a concept demonstration
 - collaborative with BYU (funded by another program)

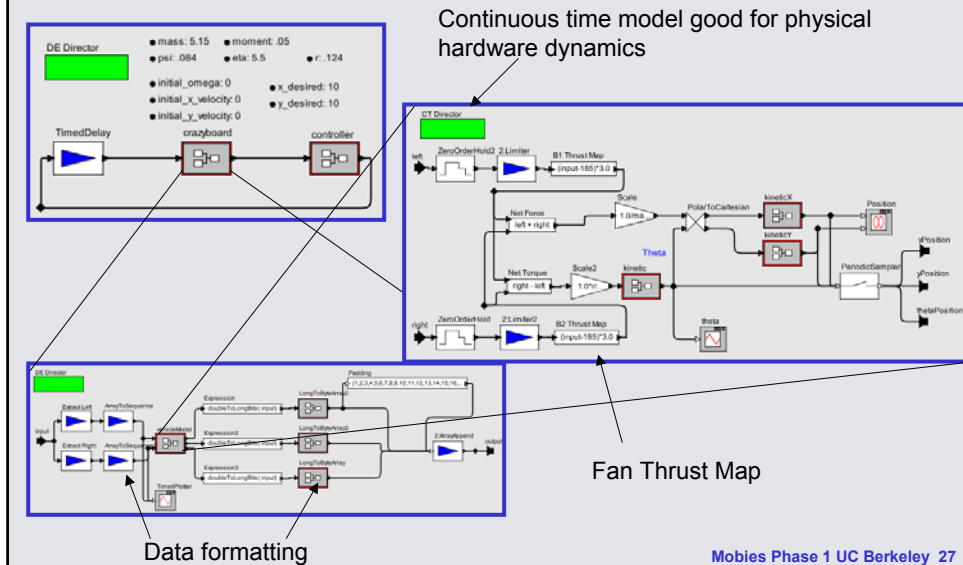
Caltech Vehicles



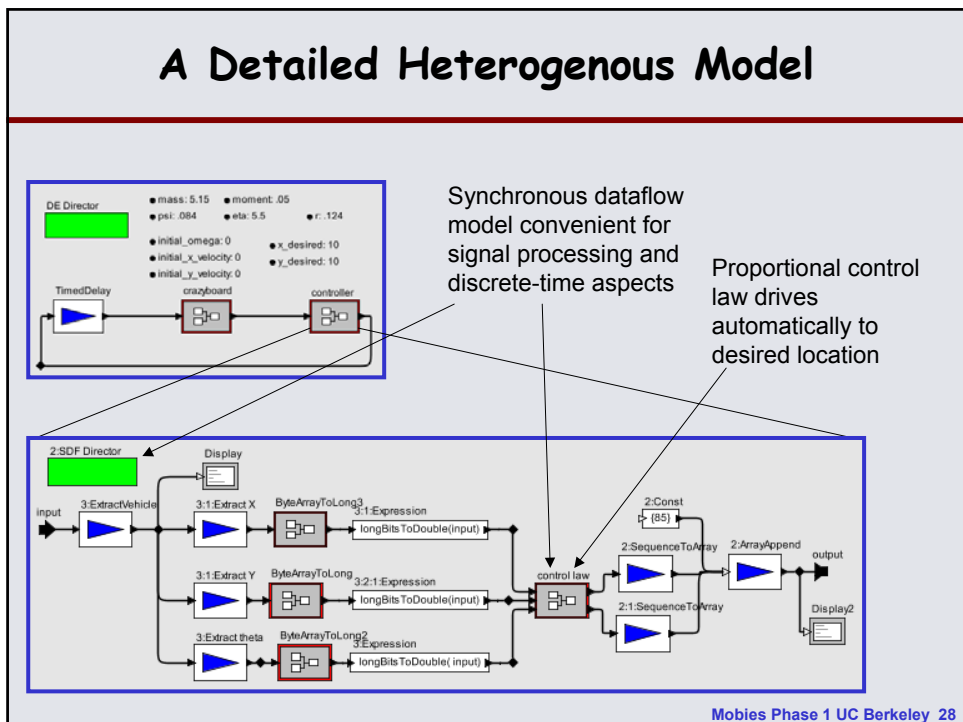
A Detailed Heterogeneous Model



A Detailed Heterogeneous Model



A Detailed Heterogenous Model

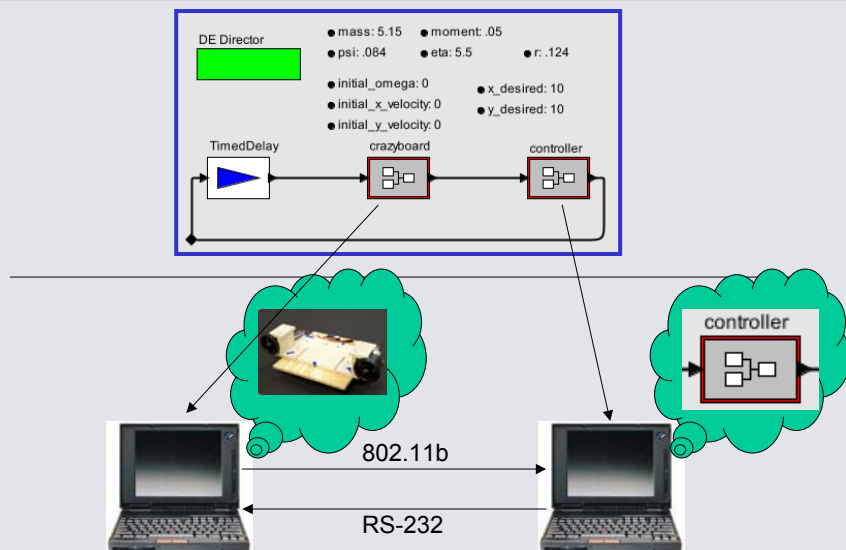


Key Observations

- Different aspects of overall system have a naturally convenient model
- Confidence in design comes from simulating heterogeneous aspects together
- Model allows validation of 'non-functional' system details (i.e. packet format, control value quantization)

Mobies Phase 1 UC Berkeley 29

Towards Implementation

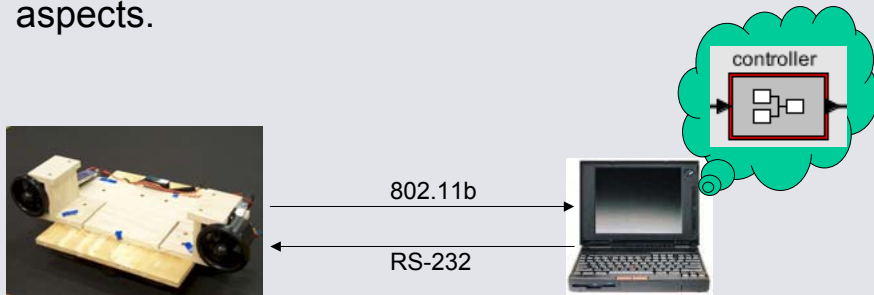


Mobies Phase 1 UC Berkeley 30

Hardware-in-the-loop

Replace hardware-true simulation model with actual vehicle.

Allows validation of hardware model aspects.

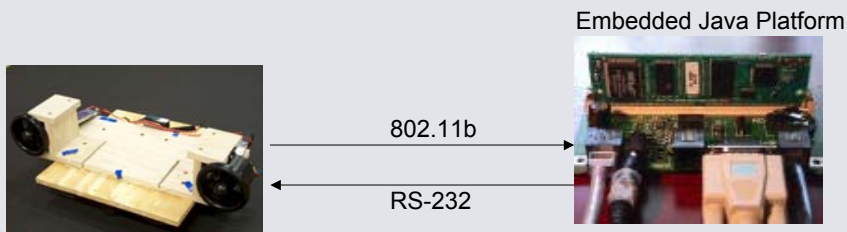


Mobies Phase 1 UC Berkeley 31

Controller Code Generation

Replace controller simulation with embedded controller.

The controller model contains all aspects of the actual embedded implementation



Mobies Phase 1 UC Berkeley 32

Directions

- Proportional control doesn't require real-time controller execution
 - Add integration term or use model-predictive control law.
 - Use timed Giotto model at toplevel of controller instead of untimed synchronous dataflow.
- Implement Softwalls algorithm on Caltech vehicles
 - Dynamics similar to 2D aircraft dynamics, but safe for experimentation.

Mobies Phase 1 UC Berkeley 33

Plans

- May 9, 2003: *Ptolemy Miniconference*
- Software radio OEP target
- Distributed models
 - CORBA integration release
 - JXTA peer-to-peer distributed applications
- HSIF
 - Resolve remaining semantics questions
 - Ensure HyVisual compliance
- Complete actor definition framework
 - define the meta-semantics for domain-polymorphic actors
- Behavioral types
 - support reflection
 - real-time properties as dependent types
- Complete code generation
 - elimination of memory management
 - 100% of test suite must pass
- Complete C code generation
 - support key subset of Java libraries
- Integrate heterogeneous code generators
 - systematize hierarchy support
 - define Java subset that generates well to C

Mobies Phase 1 UC Berkeley 34

Technology Transition Software Releases

- Ptolemy II version 2.0.1
 - Major release with many enhancements
- PtPlot version 5.2
 - Minor release with mainly 3'd party enhancements
- HyVisual version 2.2-beta
 - First domain-specific repackaging of Ptolemy II

Technology Transition Third Party Contributions (last 6 months)

- Many enhancements contributed by RIM:
 - Transition refinements
 - Higher-order components
 - Performance improvements
 - Expression language improvements
 - Matlab integration to expression language
 - Emacs integration
- Enhancements contributed by Agile Design
 - Undo/Redo
 - Port positioning
 - Icon customization
- Hardware synthesis capability from BYU
- Distributed optimization package from Spain
- Graduate class on MoCs at Virginia Tech
- Ice-cube project: paper on neutrino detection

Technology Transition Publication Summary (last 6 months)

- Published:
 - Multidimensional SDF paper (*Tr. On Signal Processing*)
- Invited:
 - Actor-Oriented Design (*J. of Circuits, Systems and Computers*)
 - Timed multitasking (*Control Systems Magazine*)
- Accepted:
 - Synchronous modeling (*Science of Computer Programming*)
 - Semantics of continuous models (*Hybrid systems workshop*)
 - Mod. & Sim. of hybrid systems (*Tr. On Mod. and Comp. Sim.*)
 - Event-driven embedded software (SAC'03)
 - Heterogeneous modeling (*IEEE Proceedings*)
- Book chapters
 - Embedded software (*Advances in Computers*)
- Conference papers:
 - Simulation of embedded control systems (a controls conference)
- Tech. Memos:
 - Behavioral Types
 - Component notations
 - Documentation for Ptolemy II v. 2.0.1