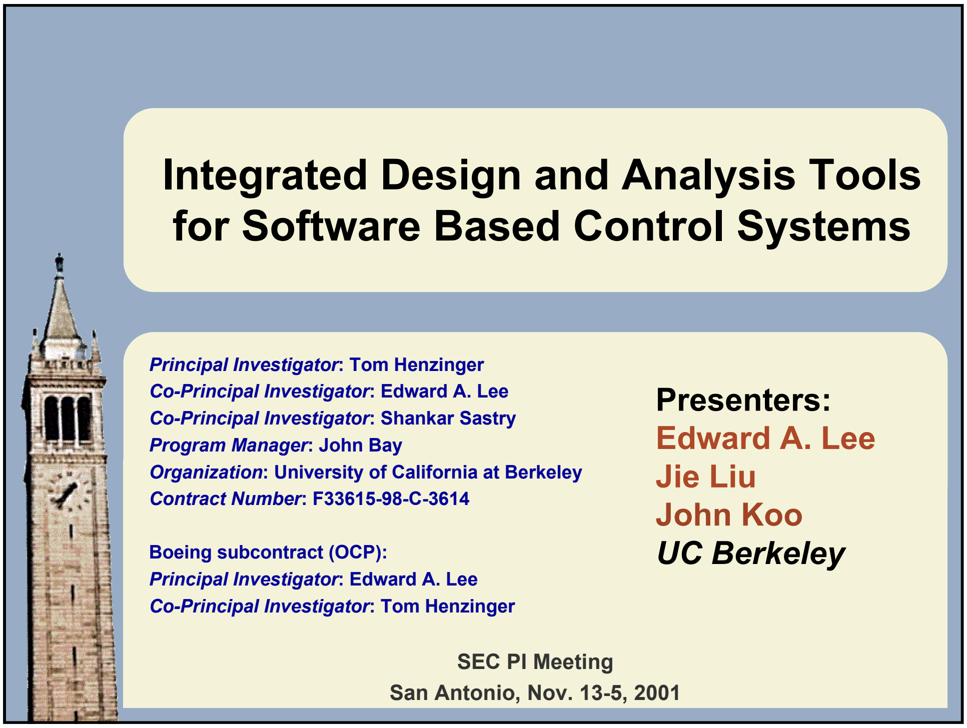


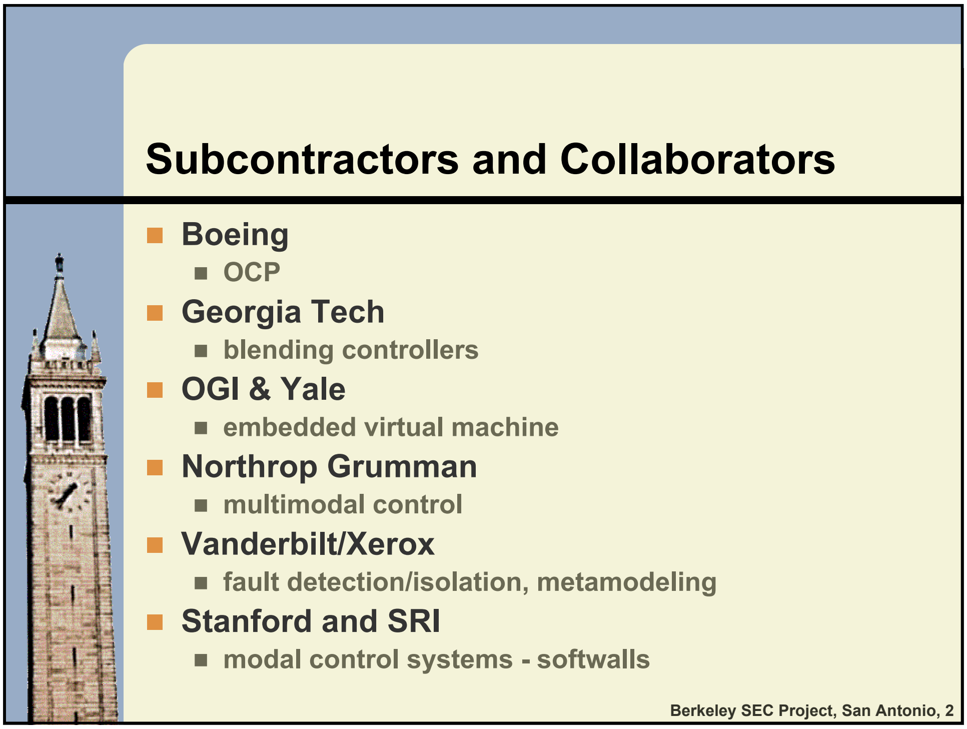
Integrated Design and Analysis Tools for Software Based Control Systems

Principal Investigator: Tom Henzinger
Co-Principal Investigator: Edward A. Lee
Co-Principal Investigator: Shankar Sastry
Program Manager: John Bay
Organization: University of California at Berkeley
Contract Number: F33615-98-C-3614

Boeing subcontract (OCP):
Principal Investigator: Edward A. Lee
Co-Principal Investigator: Tom Henzinger

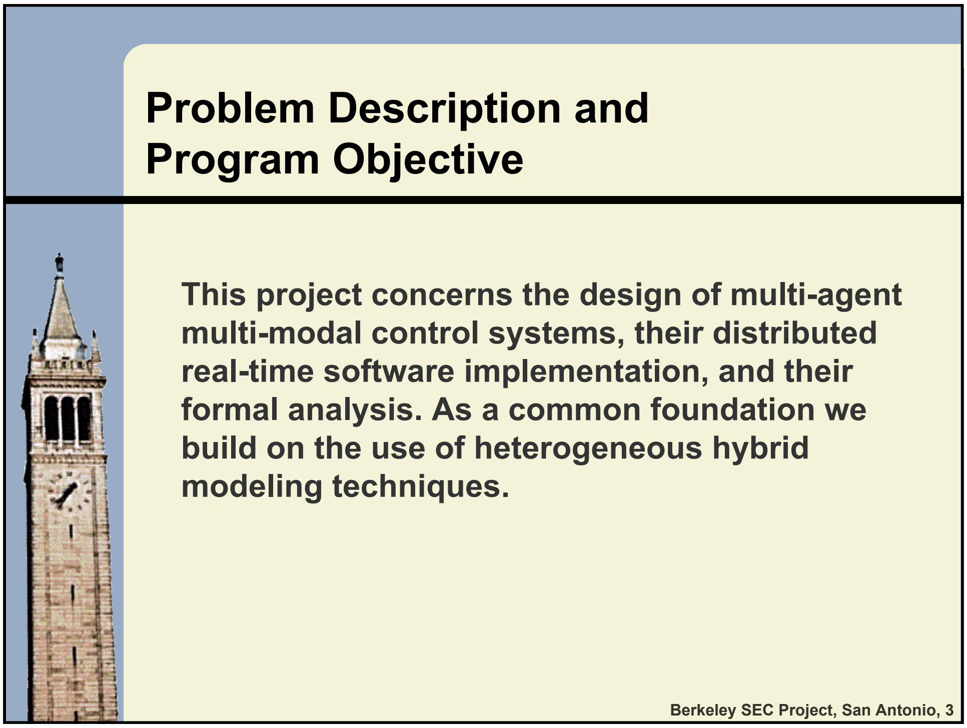
Presenters:
Edward A. Lee
Jie Liu
John Koo
UC Berkeley

SEC PI Meeting
San Antonio, Nov. 13-5, 2001



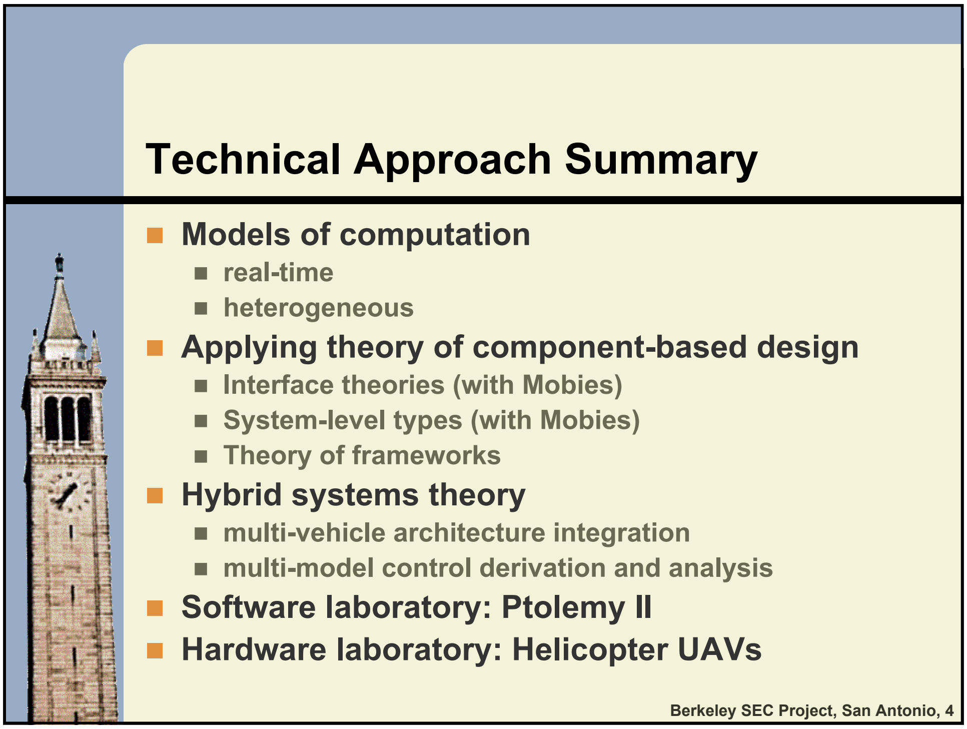
Subcontractors and Collaborators

- Boeing
  - OCP
- Georgia Tech
  - blending controllers
- OGI & Yale
  - embedded virtual machine
- Northrop Grumman
  - multimodal control
- Vanderbilt/Xerox
  - fault detection/isolation, metamodeling
- Stanford and SRI
  - modal control systems - softwalls



Problem Description and Program Objective

This project concerns the design of multi-agent multi-modal control systems, their distributed real-time software implementation, and their formal analysis. As a common foundation we build on the use of heterogeneous hybrid modeling techniques.

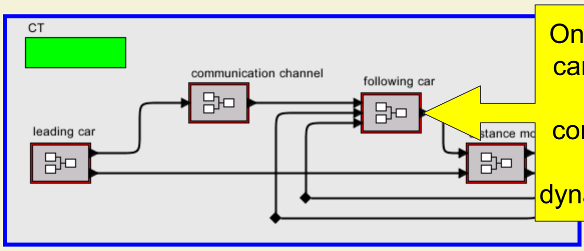


Technical Approach Summary

- Models of computation
  - real-time
  - heterogeneous
- Applying theory of component-based design
  - Interface theories (with Mobies)
  - System-level types (with Mobies)
  - Theory of frameworks
- Hybrid systems theory
  - multi-vehicle architecture integration
  - multi-model control derivation and analysis
- Software laboratory: Ptolemy II
- Hardware laboratory: Helicopter UAVs

Fault Detection, Isolation, Recovery

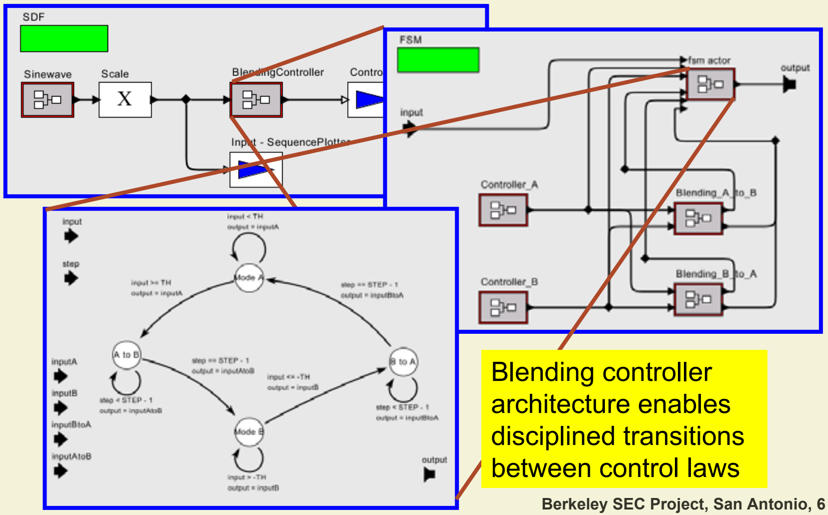
- Approach: Generalized reflection
- Demonstration: Cooperative multi-agent control



One component carries a model of another component that reflects its dynamic behavior

Reflection is a type theoretic notion of components that make available at run time models of themselves. Classically, these models represent only static type information. Our variant represents dynamics.

Blending Controllers (Collaboration with Georgia Tech)



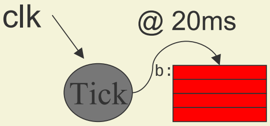
Blending controller architecture enables disciplined transitions between control laws

Embedded Virtual Machine (Collaboration with OGI and Yale)

- The embedded machine or E machine is a virtual, real-time scheduling machine
- The E machine has:
  - *ports*, *drivers*, *tasks*, and *triggers*
  - 3 key instructions + arbitrary control flow instructions
- The E machine provides a platform for generating distributed, real-time scheduling code

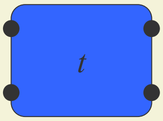
The Embedded Machine: Three Instructions

Triggering:



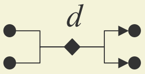
enable(g,b)

Scheduled Execution:



schedule(t)

Synchronous Execution:



call(d)

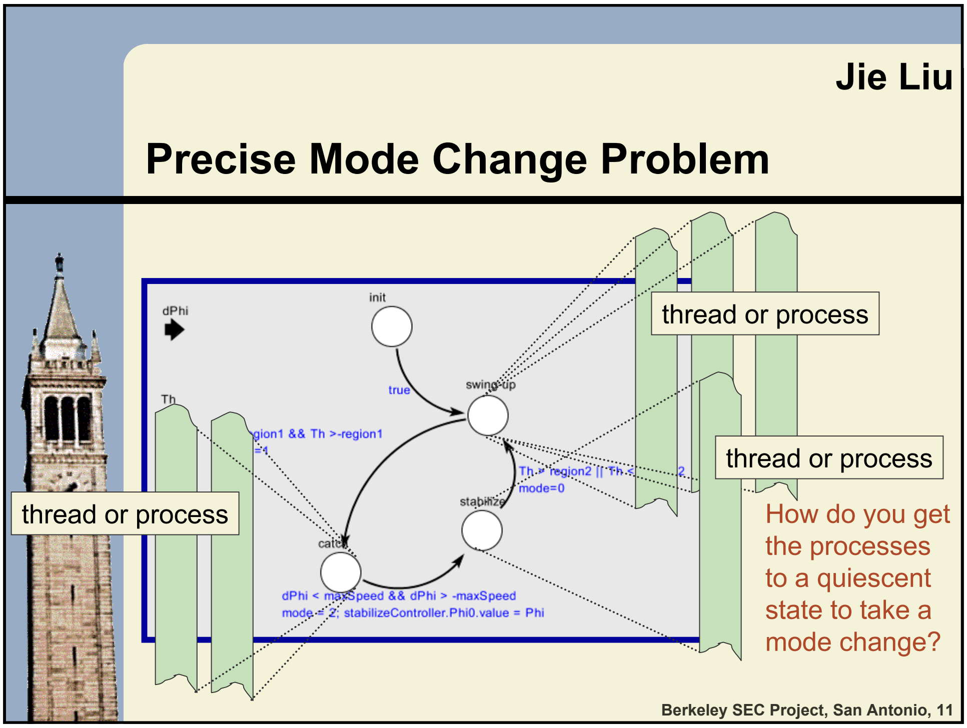
Portability, Mobility, Real-Time

- Portability: no specific hardware mapping, no specific scheduling scheme
- Mobility: dynamic upload/linking of E code; binary application code strictly separated
- Real-Time: hard real-time performance

Boeing Subcontract:
Open Control Platform - OCP

We are contributing to the future evolution of the OCP by helping to define and refine its semantics, using these semantics in hardware-in-the-loop simulation, and determining how the semantic model interoperates with others, such as FSM (for mode changes) and Giotto (for hard-real-time systems). Specific tasks include:

- Ptolemy II domains that explore OCP semantics.
- Component interfaces for real-time quality of service.
- Concurrency management.
- Solving the precise mode change problem.
- Interoperation of heterogeneous semantic models.



Jie Liu

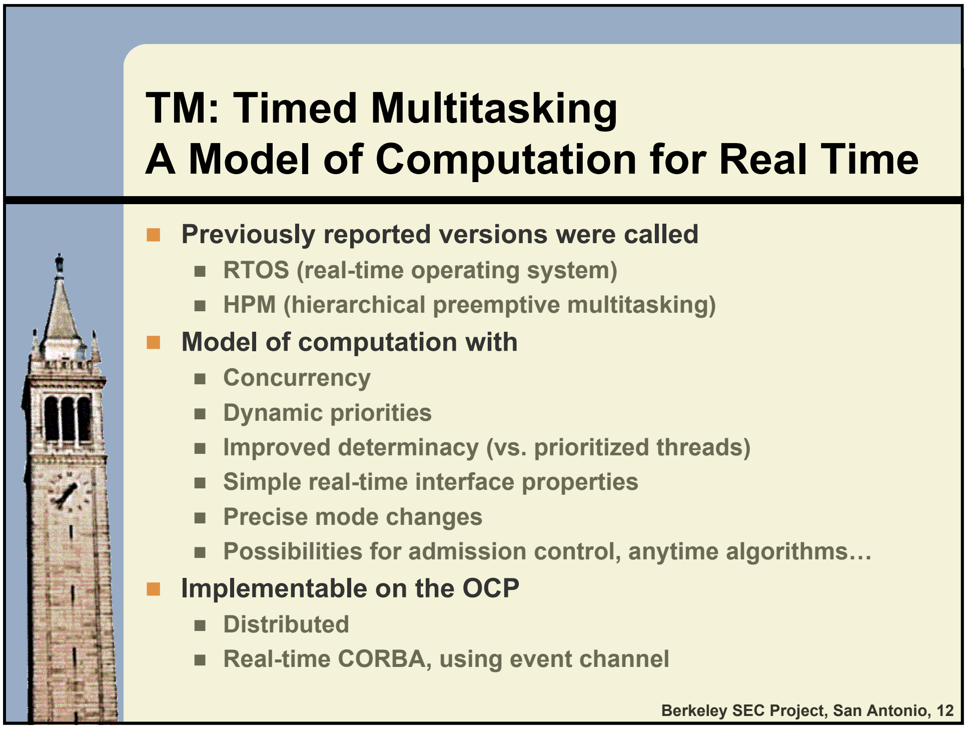
Precise Mode Change Problem

thread or process

thread or process

thread or process

How do you get the processes to a quiescent state to take a mode change?

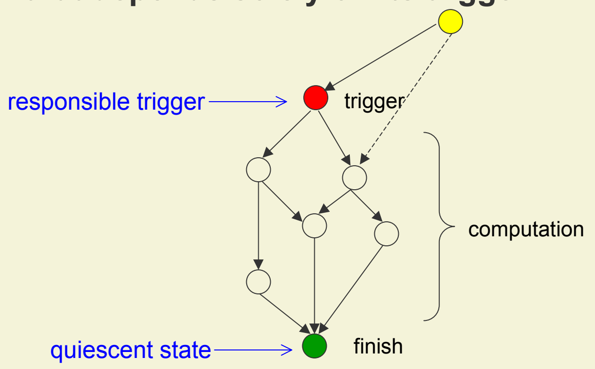


TM: Timed Multitasking A Model of Computation for Real Time

- Previously reported versions were called
  - RTOS (real-time operating system)
  - HPM (hierarchical preemptive multitasking)
- Model of computation with
  - Concurrency
  - Dynamic priorities
  - Improved determinacy (vs. prioritized threads)
  - Simple real-time interface properties
  - Precise mode changes
  - Possibilities for admission control, anytime algorithms…
- Implementable on the OCP
  - Distributed
  - Real-time CORBA, using event channel

Precise Reaction

- A ***precise reaction*** **is a finite piece of computation that depends solely on its trigger.**


responsible trigger

trigger

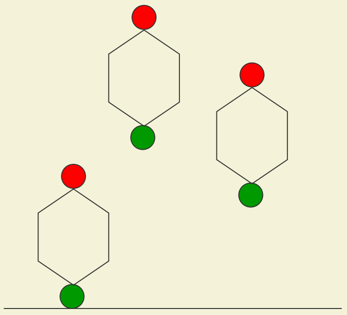
computation

quiescent state

finish

Responsible Frameworks

- A ***responsible framework*** **requests that all its components be precisely reactive and triggers these components only with responsible triggers.**



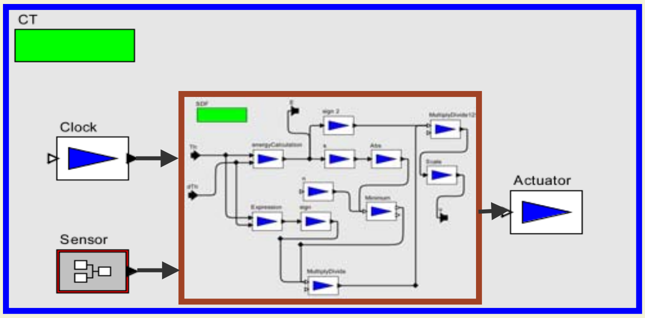
• Deadlocks can be monitored by examining triggering rules.

• A model always settles in quiescent states.

• Solves priority inversion problems in priority-based models.

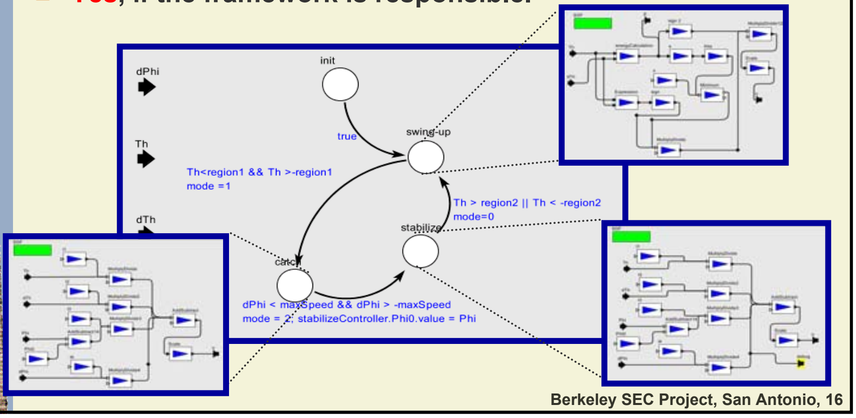
Compositional Precise Reaction

- Can we treat a composition of components as an atomic component?
- Yes, if the framework is responsible.



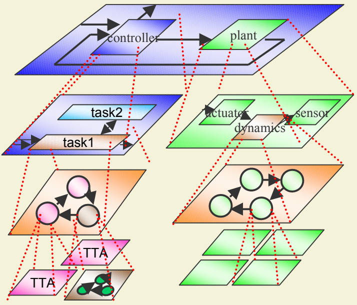
Precise Mode Change Solution

- Will the process be in a quiescent state when we do a mode change?
- Yes, if the framework is responsible.


dPhi
Th
dTh
init
true
swing-up
Th<region1 && Th >-region1
mode =1
Th > region2 || Th < -region2
mode=0
stabilize
catch
dPhi < maxSpeed && dPhi > -maxSpeed
mode = 2; stabilizeController.Phi0.value = Phi

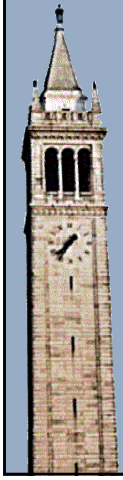
Benefits

- Composable semantics
  - arbitrarily deep hierarchies
  - heterogeneous hierarchies
- Precise mode switching
  - nest FSMs with anything else
- Real-time scheduling
  - make RT scheduling policies independent of functionality



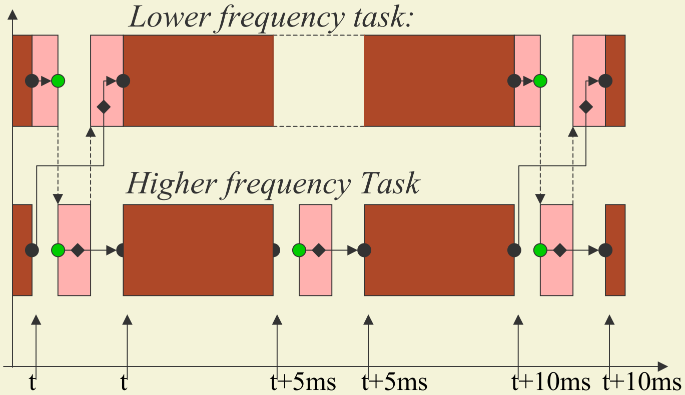
Hierarchical, heterogeneous, system-level model

Examples of Responsible Frameworks



- Dataflow with firing
  - Firing rules are responsible trigger conditions.
  - Atomic firings are precise reactions.
- Timed Multitasking
  - Tasks are either nonpreemptable or arbitrarily preemptable.
  - Event-based firing rules are responsible triggers.
  - Split-phase execution and over-run handling to guarantee timing properties.
- Giotto
  - Time are responsible triggers.
  - Well-defined communication guarantees precise reaction.
  - Tasks are arbitrarily preemptable.

Giotto – Periodic Hard-Real-Time Tasks with Precise Mode Changes



- Giotto compiler targets the E Machine
- Ptolemy II Giotto domain code generator planned

Helicopter Testbeds

- Giotto controller for Zurich helicopter written
- Giotto controller for Berkeley helicopter in progress

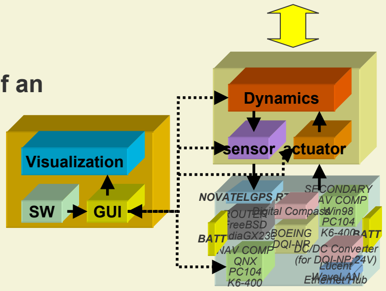
High Confidence Control Design for UAVs

John Koo

- Hybrid Control Design for Multi-Vehicle Multi-Modal Systems
  - Multi-modal controller for single vehicle
  - Coordination of multiple vehicles
- High-Confidence Hybrid Control
  - FDIR capabilities for single (envelope protection, sensor/actuator failures) and multiple vehicles (collision avoidance and conflict resolution)
- Hierarchical System Design
  - Based on parallel and serial compositions of models of computation
  - Enabling multiple vehicle corporative control
  - Implementation on OCP

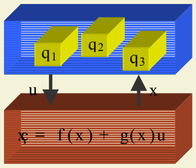
Technical Approach

- Hybrid Control design will be based on a nonlinear helicopter model and nonlinear controllers. Available for simulation in Ptolemy II and Simulink
- Hardware-In-the-Loop (HIL) simulation for architecture evaluation is currently under construction. System consists of an embedded controller and an emulator for emulating sensor/dynamics/actuator.
- Verified/Validated embedded controller will be used for controlling a R-Max helicopter.



Hierarchical Control of Multi-Modal Systems

- Given a continuous control system, a collection of *control modes* are designed

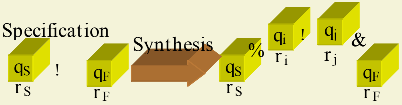


For control mode q_i;

q_i o=p : y_i = $h_i(x)$
i=p : u = $k_i(x;r_i)$
y_i ! r_i by design

Assume that r_i 2 R_i
$x(t_0)$ 2 $S_i(r_i)$ ò X_i
$x(t)$ 2 X_i; t õ t_0

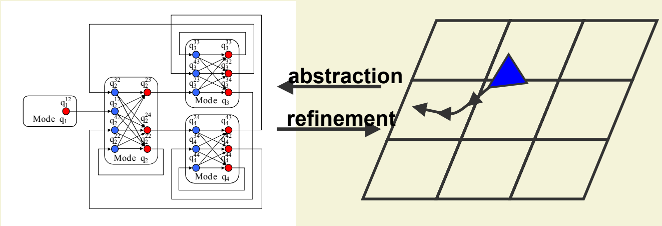
- Problem Statement of Mode Switching
  - Does there exist a finite sequence of *control modes* for satisfying a set of given reachability specifications?



  - If it does exist, can the switching conditions be determined?
    - When/ Where? Guard/Reset Synthesis
    - What Trajectory? Performance Criteria

Mode Switching Algorithm for Multi-Modal Control

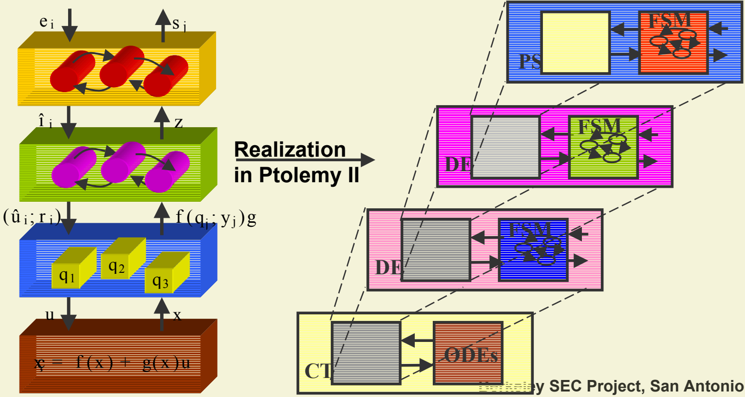
- Computation
  - Offline: Synthesis of control mode graph
    - Reachability and Intersection
  - Online: Synthesis of control switching sequence
    - Reachability on Graph



  - T. J. Koo, G. J. Pappas, and S. Sastry, “Mode Switching Synthesis for Reachability Specifications,” *Hybrid Systems: Computation and Control*, Lecture Notes in Computer Science, Springer, 2001.

Hierarchical Component-Based Design

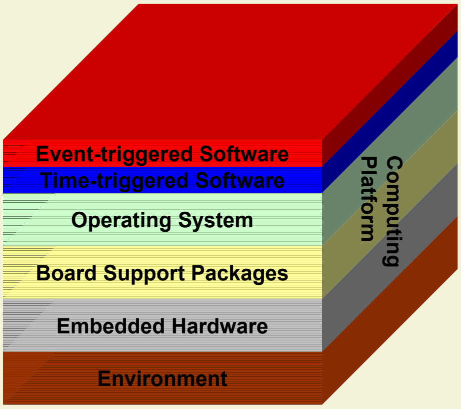
- Hierarchical nesting of compositions of discrete and continuous components
- At each level of the hierarchy, a Model of Computation (MoC) governs the behaviors and interactions of components



Realization in Ptolemy II

Implementing a Design in Embedded Software

- Question: How to guarantee safety of the embedded system?



Event-triggered Software

Time-triggered Software

Operating System

Board Support Packages

Embedded Hardware

Environment

Computing Platform

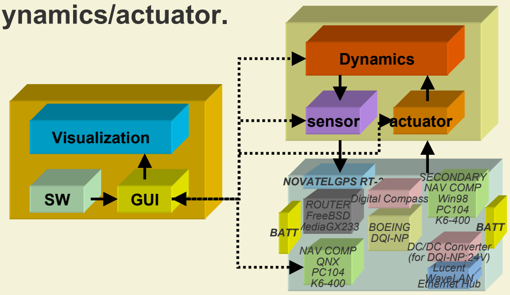
- Our Solution:

  at the level closest to the environment under control, the embedded software needs to be time-triggered for guaranteed safety;

  at higher levels, an asynchronous hybrid controller design is required.

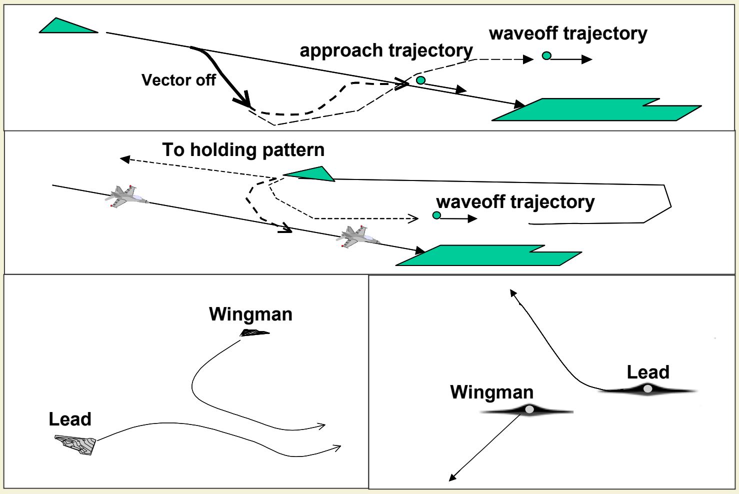
Ongoing Work

- Hardware-In-the-Loop (HIL) simulation for architecture evaluation is currently under construction. System consists of an embedded controller and an emulator for sensor/dynamics/actuator.



- Verified/Validated embedded controller will control an R-Max helicopter.

Candidate Real-life Applications (with Northrop-Grumman)



Project Tasks/Schedule/Status

- Demos done
  - Multi-modal helicopter control model (hybrid system)
  - Fault detection/isolation based on generalized reflection
  - Blending controller (with Georgia Tech)
  - Publish & subscribe using Jini and JavaSpaces
  - Giotto helicopter control for Zurich helicopter
  - Precise mode changes using TM domain
  - Multi-modal distributed control (with Lego robots)
- Fundamental contributions
  - Framework theory
    - responsible frameworks
    - precise reactions/mode changes
    - managing heterogeneity
    - models of computation
  - Timed multitasking model of computation
  - Giotto time-triggered model of computation
  - Multimodal control framework
  - Controller synthesis for safety properties

Next Milestones

- Future milestones
  - Giotto helicopter control for Berkeley helicopters
  - Hardware-in-the-loop simulation
  - CORBA/OCP event channel interface to the TM domain
  - OCP E Machine realization
  - E Machine realizations running hard-real-time code
  - FDIR in hybrid controllers for single/ multiple vehicles
  - Multi-vehicle formation flight
- Anticipated fundamental contributions
  - Just watch!

Technology Transition/Transfer

- Classic tech transfer strategy:
  - Copyright
  - Retain intellectual property & leverage the profit motive
- Radical tech transfer strategy:
  - Copyleft
  - Distribute freely & impose your ideology on others
- Berkeley tech transfer strategy:
  - Copycenter
  - Take it to the copy center & copy as much as you like.
- Success of this model:
  - Many companies have brought Berkeley research results into the marketplace.

Technology Transition/Transfer – Near-Term Plans

- E Machine pilot implementations will show how to
  - Isolate designers from RTOS platforms
  - Get a coherent semantics in the run-time environment
- Giotto model of computation will show how to
  - Build hard-real-time, periodic, multimodal models
  - Specify real-time requirement (vs. infer real-time behavior)
- TM model of computation will show how to
  - Build priority-driven multitasking with precise mode changes
- Ptolemy II version 2.0 release will show how to
  - Get precise mode changes in a real-time multitasking context
  - Realize multi-modal multi-agent hybrid systems
  - Realize blending controllers
- Helicopter control models will show how to
  - Hierarchically build autonomous multi-vehicle control systems with hybrid control methods.
  - Work with Northrop-Grumman to transfer methods.

Program Issues – Employing SEC Technology

- Homeland defense – softwalls
  - carry on-board 3-D database with “no-fly-zones”
  - enforce in the on-board avionics, based on localization
  - non-networked, non-hackable
  - hybrid, modal controller in embedded software