Overview of Research at CHESS
Center for Hybrid and Embedded Software Systems

Stavros Tripakis
Associate Researcher
UC Berkeley
stavros@eecs.berkeley.edu

Presented to IHI Delegates
Feb 20, 2013
Berkeley, CA

University of California at Berkeley

UC Berkeley has one of the best public engineering schools in the world.
Overview of EECS at Berkeley

Electronic Design Automation
Algorithms and techniques to support computer-aided design and optimization of complex hardware and software systems.

Embedded Software Systems
Models of computation, specification languages, real-time systems, hardware and software synthesis and compilation for electronic systems.

Modeling and Verification
Models of hardware and software systems together with analysis techniques that identify design flaws, performance problems, and vulnerabilities.
CHESS: Center for Hybrid and Embedded Software Systems

Principal Investigators
- Thomas Henzinger (EPFL)
- Edward A. Lee (Berkeley)
- Alberto Sangiovanni-Vincentelli (Berkeley)
- Shankar Sastry (Berkeley)
- Janos Szilagyi (Vanderbilt)
- Claire Tomlin (Berkeley)

Executive Director
- Christopher Brooks

Associated Faculty
- David Auslander (Berkeley, ME)
- Ahmad Bahai (Berkeley)
- Ruzena Bajcsy (Berkeley)
- Gautam Biswas (Vanderbilt)
- Ras Bodik (Berkeley, CS)
- Bella Bollobas (Memphis)
- Karl Hedrick (Berkeley, ME)
- Gabor Karsai (Vanderbilt)
- Kurt Keutzer (Berkeley)
- George Necula (Berkeley, CS)
- Koushik Sen (Berkeley, CS)
- Sanjit Seshia (Berkeley)
- Jonathan Sprinkle (Arizona)
- Masayoshi Tomizuka (Berkeley, ME)
- Pravin Varaiya (Berkeley)

Some Research Projects
- Precision-timed (PRET) machines
- Distributed real-time computing
- Systems of systems
- Theoretical foundations of CPS
- Hybrid systems
- Design technologies
- Verification
- Intelligent control
- Modeling and simulation

Applications
- Air traffic control
- Avionics
- Automotive
- Building systems
- Factory automation
- Instrumentation
- Medical systems
- Process control
- Synthetic biology
- Test & measurement

Hybrid Systems
Where it started

A model of a spring-mass system with collisions, modeled in Ptolemy II:

Consider the velocity of each mass. Is it continuous? What about the acceleration?
Hybrid automata = state machines + differential equations

Cyber-Physical Systems (CPS)

Where it is going

CPS: Orchestrating networked computational resources with physical systems.
Computers as parts of embedded systems

“~98% of the world’s processors are not in PCs but are embedded”

“a premium car today has:
- ~80 computers (ECUs – Electronic Control Units)
- ~100 million lines of code
- ~2km of wiring (CAN bus, other networks ...)

Embedded system languages & tools

Key concepts:
- reactive behavior
- concurrency
- timing
- I/O
- ...

Key capabilities:
- simulation
- code generation
- verification
These modeling languages of today will become the **system-programming** languages of tomorrow

Rich input languages:
- concurrency, time,
- robustness, reliability,
- energy, security, …

Powerful analyses:
- model-checking, WCET analysis, schedulability,
- performance analysis, reliability analysis, …

Complex execution platforms:
- networked, distributed,
- multicore, …

---

**Model-Based Design**

**How to describe what we want?**

**How to be sure that this is what we want?**

**How to build it? Automatically Correct-by-construction**
Is Model-Based Design a realistic vision?

Tripakis, Berkeley 13

Instance of a successful and model-based design flow

Tripakis, Berkeley 14
Platform-based design

Component-based design

Alberto Sangiovanni-Vincentelli

Tripakis, Berkeley 15

Tripakis, Berkeley 16
Agenda

- Some research activities going on at CHESS: this talk, until 10:30am (Stavros)
- Tour of CHESS facilities, discuss with students and researchers, look at selected projects: 10:30am – noon
  - 10:35am - 10:50am: David Broman, Michael Zimmer – PRET
  - 10:50am - 11:05am: Patricia Derler - Ptides
  - 11:05am - 11:20am: Alex Donze - Hybrid system verification
  - 11:20am - 11:35am: Ben Zheng - Terraswarm and localization
  - 11:35am - 11:50am: Mehdi Maasoumy - Smart/green buildings
  - 11:50am - noon: Sayak Ray - HW verification and synthesis

Some of the CPS Research in Chess

- **Foundations**: Heterogeneous modeling with actors.
- **Bottom up**: Embedded processors (PRET).
- **Top down**: Distributed real-time systems (PTIDES).
- **Holistic**: Scalable model-based design.
The Ptolemy project

The Ptolemy project studies modeling, simulation, and design of concurrent, real-time, embedded systems. The focus is on assembly of concurrent components. The key underlying principle in the project is the use of well-defined models of computation that govern the interaction between components. A major problem area being addressed is the use of heterogeneous mixtures of models of computation. A software system called Ptolemy II is being constructed in Java, and serves as the principal laboratory for experimentation.

Contributors to Ptolemy II

Principal Authors
- Christopher Brooks
- Dal Bul
- Chamberlain Feng
- John Davis, III
- Patricia Dierk
- Thomas Huntington Feng
- Mudi Goel
- Rowland Johnson
- Bing Lee
- Edward Lee
- Alan Lickly
- Jie Liu
- Xiaojun Liu
- Lukito Mulladi
- Stephen Neuendorffer
- John Remke
- Neil Smyth
- Jeff Taylor
- Yu Hong Xiong
- Huiyong Zheng
- Gang Zhou

Other Contributors
- Jim Amerongen
- Vincent Arnoudt
- Kyungtae Baik
- Peter Balcerz
- Chad Baur
- Frederic Biostinger
- Raymond Cardinal
- Janette Cardoso
- Adam Casado
- Christine Casavant
- Chris Chang
- Albert Chen
- Chiehong Patrick Cheng
- Elaine Cheung
- Colin Cocks
- Brice Dussault
- Pedro Dominguez
- William Douglas
- Johan Eer
- Thomas Huntington Feng
- Toivo Pirkola
- Tszr Fungso
- Shaank-Shaoyu Forbes
- Jaiyako Fuente
- Gonzalo Galicia
- Ben Hanowitz
- Heloise Hoe
- Ebrah Jaagar
- Jon Jannack
- Zofia Kaminska
- Bart Kleinheurs
- Christoph Meyer Kirsch
- Sanghyun Kohl
- Vindy Kohlan
- Robert Krogser
- Daniel Lazaar
- David Lee
- Man-i (Jackie) Leung
- Michael Leung
- John Li
- Isaac Liu
- Andrew Mihal
- Etienne Molokovos
- Alexandre Mouren
- Mike Noll
- Sarah Packman
- Shankar Ras
- Bert Roders
- Rajkumar Reddy
- Atanasia Ricci
- Sonia Sahh
- Joaquin S. Martinez
- Michael Shimian

Lee, Berkeley 4

Tripakis, Berkeley 19
Heterogeneous modeling

- One system, many models!
- Different parts of the same system typically modeled using different types of models.
- How to combine these different models?
- How to reason about the system as a whole?

Examples of heterogeneous models

- Control systems = discrete + continuous dynamics

Simulink
Copyright The Mathworks

Tripakis, Berkeley 21

Tripakis, Berkeley 22
Hierarchical Multimodeling
Hierarchical compositions of graphical models in Ptolemy II.

Ptolemy = Actors + Directors + …

- Actors: basic functionality (“atomic” blocks)
- Directors: simulation engines
  - Give meaning to the composition of actors
  - Different engines implement different models of concurrency (synchronous, asynchronous, discrete, continuous, …)
  - (almost) Arbitrary hierarchical compositions
- + GUI + type system + …
Ptolemy II: a Laboratory for Actor-Oriented Models of Computation

Concurrency management supporting dynamic model structure.

Director from an extensible library defines component interaction semantics or "model of computation."

Extensible, behaviorally-polymorphic component library.

Type system for transported data.

Visual editor supporting an abstract syntax.

Actors
Actors

- Ptolemy has a library of predefined actors
- They are all Java classes that implement a standard interface called “executable”

Some actors

Directors
Hierarchical Multimodeling
Hierarchical compositions of graphical models in Ptolemy II.

Models of Computation Implemented in Ptolemy II

- CI – Push/pull component interaction
- Click – Push/pull with method invocation
- CSP – concurrent threads with rendezvous
- Continuous – continuous-time modeling with fixed-point semantics
- 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 – 3-D graphics
- PN – process networks
- Rendezvous – extension of CSP
- SDF – synchronous dataflow
- SR – synchronous/reactive
- TM – timed multitasking
- ...
Ptolemy II Software Architecture
Built for Extensibility

Ptolemy II packages have carefully constructed dependencies and interfaces.

Tripakis, Berkeley 31

Brief Ptolemy demo

A car-tracking controller, modeled in Ptolemy II.

Tripakis, Berkeley 32
Some of the CPS Research in Chess

- **Foundations**: Timed computational semantics.
- **Bottom up**: Embedded processors (PRET).
- **Top down**: Distributed real-time systems (PTIDES).
- **Holistic**: Scalable model-based design.

**Bottom Up: Embedded Processors**

**Precision-Timed (PRET) Machines**

*Make temporal behavior as important as logical function.*

Timing precision with performance: Challenges:
- Memory hierarchy (scratchpads?)
- Deep pipelines (interleaving?)
- ISAs with timing (deadline instructions?)
- Multicore (dedicated I/O & real-time processors?)
- Predictable memory management (Metronome?)
- Languages with timing (discrete events? Giotto?)
- Predictable concurrency (synchronous languages?)
- Composable timed components (actor-oriented?)
- Precision networks (TTA? Time synchronization?)

Some of the CPS Research in Chess

- **Foundations**: Timed computational semantics.
- **Bottom up**: Embedded processors (PRET).
- **Top down**: Distributed real-time systems (PTIDES).
- **Holistic**: Scalable model-based design.

PTIDES: Programming Temporally Integrated Distributed Embedded Systems

*Distributed execution under DE semantics, with “model time” and “real time” bound at sensors and actuators.*
Some of the CPS Research in Chess

- **Foundations**: Timed computational semantics.
- **Bottom up**: Embedded processors (PRET).
- **Top down**: Distributed real-time systems (PTIDES).
- **Holistic**: Scalable model-based design.

From multi-modeling to *multi-view* modeling
Viewpoints, views and models

The view is what you see

The viewpoint is where you look from

Real-world/System

Models

Abstractions

Martin Törngren

Tripakis, Berkeley 39

Multiple viewpoints

<table>
<thead>
<tr>
<th>Stakeholder/Role</th>
<th>Concerns</th>
<th>Analysis/Synthesis</th>
<th>Model Characteristics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Electrical engineer/hardware architecture</td>
<td>ECU interfaces and EMC</td>
<td>Electrical load and tests</td>
<td>Logic, continuous, and FEM</td>
</tr>
<tr>
<td>Software engineer (body area)</td>
<td>Logics of functionality and Reliability</td>
<td>Simulation of behavior</td>
<td>Discrete-event</td>
</tr>
<tr>
<td>Quality engineer</td>
<td>ECU packaging, geometry, and fitting</td>
<td>Life-time prediction and FMEA</td>
<td>Stochastic and logic (e.g., failure analysis)</td>
</tr>
<tr>
<td>Mechanical engineer</td>
<td>ECU packaging, geometry, and fitting</td>
<td>Cable length and geometry alignment</td>
<td>2D and 3D mechanical</td>
</tr>
<tr>
<td>Cost controller</td>
<td>Product cost</td>
<td>Profitability and sensitivity analysis</td>
<td>Economical and uncertainty explicit</td>
</tr>
<tr>
<td>Integration engineer</td>
<td>Verification communications and distributed functions</td>
<td>Testing! Automation of tests and generation of test documentation</td>
<td>Discrete event (test cases) and logical structures (e.g., configurations)</td>
</tr>
<tr>
<td>Safety engineer</td>
<td>System safety</td>
<td>FTA and FMEA</td>
<td>Logic, discrete-event, and stochastic</td>
</tr>
<tr>
<td>Control system engineer</td>
<td>Performance and robustness, and disturbances</td>
<td>Behavior simulation, robustness analysis, and controller synthesis</td>
<td>Continuous-time, discrete-time, and discrete-event</td>
</tr>
<tr>
<td>Thermoanalyst</td>
<td>Temperature</td>
<td>Heat transfer</td>
<td>FEM, etc.</td>
</tr>
</tbody>
</table>
Multiple viewpoints

A framework to support stakeholders - focus on modeling
Multiple views (projections), with partially overlapping and related content

Multi-View Modeling:
Distinct and separate models of the same system are constructed to model different aspects of the system.

Functional model in Statecharts
Functional model in Ptolemy II
Deployment model in Ptolemy II
Verification model in SMV
Reliability model in Excel

This example is a test case for a collaborative project with Lockheed-Martin

Tripakis, Berkeley 43
Tripakis, Berkeley 44
Multi-view modeling

- Example concrete problem:
  - How to maintain consistency between AADL Behavior annex and Error annex models?

Design Contracts

- Digital controller
- Plant
- Embedded system platform

Requirements
- Timing
- Computation
- Accuracy
- System level requirements:
  - Cost
  - Dependability
  - Performance
  - Flexibility

Constraints
- Memory
- Power

Martin Törngren
Example timing contract for control

Bounds on the admissible variations of period and StA delay respectively

Sampling instant $k$

Actuation instant $k$

Sampling to Actuation delay

Compositional Model-Based Design

How to create large models from libraries of submodels?

How to analyze submodels separately and derive global system properties?

How to implement different parts of the system independently?
Substitutability

\[ v \in [v_{\min}, v_{\max}] \]

latency \leq 10ms

Tripakis, Berkeley 49

Substitutability

\[ v \in [v_{\min}, v_{\max}] \]

latency \leq 10ms

Z

Tripakis, Berkeley 50
Substitutability

How to ensure properties are preserved?

\[ v \in [v_{\text{min}}, v_{\text{max}}] \]

Interface theories = **behavioral** type theories

**Type error!**

\[ v \in [v_{\text{min}}, v_{\text{max}}] \]

Which type theories for safety & performance properties?
Research as a Business

Our Business Model:

Maximize the impact of our work, and the rest follows

Chess Industrial Partnerships

- Close interaction between academic research and industrial experience, often involving frequent deep technical interactions.

- Facilitate the creation and transfer of modern, "new economy" software technology methods and tools to "old economy" market sectors.

- Focus on industries where embedded software plays a central role, such as:
  - aerospace
  - automotive
  - Instrumentation, test, and measurement
Options for Funding Research

- Gifts
- Quasi-gifts (CADCAM, BEECSA, ...)
- Center memberships
- Contracts

Center memberships have proved far more effective than the alternatives.

Intellectual Property in CHESS

Commitment is:

- Software will be open source
- Patents will be rare
IP Principles

- Address researcher concerns
  - Maximize the impact of the work
  - Freedom to publish papers
  - Freedom to release open source software
  - No obligation to patent results
  - Ability to patent results
  - Minimal obstacles to commercialization

- Address company concerns:
  - Value for the money
  - Protection from being “locked out” of research results
  - Viable VIF agreements
  - Desire to “expense” contributions

Intellectual Property Rights

*From Appendix C of the CHESS Agreement*

- “The objective of CHESS is to maximize the impact of its research. To achieve this, CHESS will maintain an open atmosphere that encourages early and frequent publication and other public dissemination of research results. Software will primarily be released using an open source license such as the Berkeley Software Distribution (BSD) License. Selected software, such as those subject to third party obligations, may be released under different licenses.”
- “Patents are expected to be rare. . .”
The BSD License: Used for most software

Copyright (c) YEAR The Regents of the University of California.
All rights reserved.

Permission is hereby granted, without written agreement and without license
or royalty fees, to use, copy, modify, and distribute this software and its
documentation for any purpose, provided that the above copyright notice and
the following two paragraphs appear in all copies of this software.

IN NO EVENT SHALL THE UNIVERSITY OF CALIFORNIA BE LIABLE TO ANY PARTY FOR
DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES ARISING OUT
OF THE USE OF THIS SOFTWARE AND ITS DOCUMENTATION, EVEN IF THE UNIVERSITY OF
CALIFORNIA HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

THE UNIVERSITY OF CALIFORNIA SPECIFICALLY DISCLAIMS ANY WARRANTIES,
INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND
FITNESS FOR A PARTICULAR PURPOSE. THE SOFTWARE PROVIDED HEREUNDER IS ON AN
"AS IS" BASIS, AND THE UNIVERSITY OF CALIFORNIA HAS NO OBLIGATION TO PROVIDE
MAINTENANCE, SUPPORT, UPDATES, ENHANCEMENTS, OR MODIFICATIONS.

Spinoffs of Ptolemy Software

- **Agilent ADS**: Leading design software for analog, RF, and mixed-signal design,
  primarily targeted to wireless systems development, based on Ptolemy Classic.
- **Cal actor design language**, adopted by MPEG for specification of coding
  standards, and used by Thales, Xilinx, and others for FPGA and multicore software
  design.
- **iSencia Passerelle** is based on Ptolemy II and is used to prepare experiments for
  the Soleil synchrotron.
- **Kepler**: A System for Scientific Workflows, is a cross-project collaboration to
  develop open source tools for Scientific Workflows and is based on the Ptolemy II.
- **Mirabilis Design VisualSim**, built on top of Ptolemy II, does performance analysis
  and system architecture, rapid system modeling and hardware/software tradeoff
  analysis.
- **ML Design Technologies’ MLDesigner**, is a platform that leverages Ptolemy
  Classic for modeling and analyzing the architecture, function and performance of
  high level system.
- **And more**: Boeing, Bosch, Cadence, Lockheed Martin, Research in Motion,
  Thales (ArrayOL), VPI Systems, White Eagle Technologies, etc.
**CHESS Industrial Membership Levels**

<table>
<thead>
<tr>
<th>Membership Level</th>
<th>Benefits</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Affiliate</strong> &gt; $75k/year</td>
<td>- Invitation to periodic reviews of CHESS.</td>
</tr>
<tr>
<td></td>
<td>- Access to selected internal CHESS websites.</td>
</tr>
<tr>
<td></td>
<td>- Access to publications, reports and presentations by CHESS researchers.</td>
</tr>
<tr>
<td></td>
<td>- Access to students and faculty in CHESS.</td>
</tr>
<tr>
<td></td>
<td>- An annual research report of the activities of CHESS.</td>
</tr>
<tr>
<td></td>
<td>- Advance notice of intellectual property created by CHESS.</td>
</tr>
<tr>
<td></td>
<td>- Intellectual property access as defined in the agreement. (See Appendix C(1))</td>
</tr>
<tr>
<td><strong>Small or Minority-Owned Business &gt; $10k/year</strong></td>
<td>All of the benefits of an Affiliate</td>
</tr>
<tr>
<td><strong>Partner</strong> &gt; $150k/year</td>
<td>All of the benefits of an Affiliate, plus the following:</td>
</tr>
<tr>
<td></td>
<td>- Opportunity to place visitors, as Visiting Industrial Fellows (VIF) at the University.</td>
</tr>
<tr>
<td></td>
<td>- Selected early access to software developed by CHESS.</td>
</tr>
<tr>
<td><strong>Premium Partner</strong> &gt; $300k/year</td>
<td>All of the benefits of a Partner, plus the following:</td>
</tr>
<tr>
<td></td>
<td>- Intellectual property access as defined in the agreement.</td>
</tr>
<tr>
<td></td>
<td>- Upon request, an annual private research review meeting at the University or at a mutually agreeable site.</td>
</tr>
</tbody>
</table>

**Conclusion**

Chess has a rich portfolio of research projects and an established track record of working effectively with industry to maximize the impact of the work.
Agenda

- Some research activities going on at CHESS: this talk, until 10:30am (Stavros)
- Tour of CHESS facilities, discuss with students and researchers, look at selected projects: 10:30am – noon

- 10:35am - 10:50am: David Broman, Michael Zimmer – PRET
- 10:50am - 11:05am: Patricia Derler - Ptides
- 11:05am - 11:20am: Alex Donze - Hybrid system verification
- 11:20am - 11:35am: Ben Zheng - Terraswarm and localization
- 11:35am - 11:50am: Mehdi Maasoumy - Smart/green buildings
- 11:50am - noon: Sayak Ray - HW verification and synthesis