The Case for Timing-Centric Distributed Software

Edward A. Lee

Robert S. Pepper Distinguished Professor
UC Berkeley

With Slobodan Matic, Sanjit A. Seshia, and Jia Zou

Invited Talk
2nd International Workshop on Cyber-Physical Systems (WCPS 2009)

June 22, 2009
Montreal, Quebec, Canada
Context of my work: Chess: Center for Hybrid and Embedded Software Systems

Board of Directors
- Edward A. Lee
- Alberto Sangiovanni-Vincentelli
- Shankar Sastry
- Claire Tomlin

Executive Director
- Christopher Brooks

Other key faculty at Berkeley
- Dave Auslander
- Ruzena Bajcsy
- Raz Bodik
- Karl Hedrick
- Kurt Keutzer
- George Necula
- Masayoshi Tomizuka
- Pravin Varaiya

This center, founded in 2002, blends systems theorists and application domain experts with software technologists and computer scientists.

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
- Building systems
- Automotive
- Synthetic biology
- Medical systems
- Instrumentation
- Factory automation
- Avionics
Cyber-Physical Systems (CPS): Orchestrating networked computational resources with physical systems

- **Automotive**: E-Corner, Siemens
- **Military systems**: Daimler-Chrysler
- **Power generation and distribution**: Courtesy of General Electric
- **Building Systems**: Courtesy of General Electric
- **Avionics**: Courtesy of Kuka Robotics Corp.
- **Telecommunications**: Courtesy of General Electric
- **Transportation (Air traffic control at SFO)**: Courtesy of General Electric
- **Factory automation**: Courtesy of Kuka Robotics Corp.
- **Innovation**: (Soleil Synchrotron)
CPS Example – Printing Press

- **High-speed, high precision**
  - Speed: 1 inch/ms
  - Precision: 0.01 inch
    - Time accuracy: 10us

- **Open standards (Ethernet)**
  - Synchronous, Time-Triggered
  - IEEE 1588 time-sync protocol

- **Application aspects**
  - local (control)
  - distributed (coordination)
  - global (modes)

Bosch-Rexroth
Object Oriented vs. Actor Oriented

The established: Object-oriented:

- Class name
- Data
- Methods

What flows through an object is sequential control.

Things happen to objects.

The alternative: Actor oriented:

- Actor name
- Data (state)
- Parameters
- Ports

What flows through an object is evolving data.

Actors make things happen.
Some Actor-Oriented Influences

- BIP [Basu, Bozga, Sifakis 2006]
- Colif [Jerraya et al. 2001]
- Esterel [Berry et al. 1992]
- ForSyDe [Sander, Jantsch 2004]
- FunState [Thiele, Ernst, Teich, et al. 2001]
- Giotto [Henzinger et al. 2001]
- HetSC [Herrera, Villar 2006]
- LabVIEW [Kodosky et al. 1986]
- Lustre [Halbwachs, Caspi et al. 1991]
- Metropolis [Goessler, Sangiovanni-Vincentelli et al. 2002]
- Ptolemy Classic [Buck, Ha, Messerschmitt, Lee et al. 1994]
- Ptolemy II [Eker, Janneck, Lee, et al. 2003]
- RTComposer [Alur, Weiss 2008]
- SCADE [Berry et al. 2003]
- SDL [Various, 1990s]
- Signal [Benveniste, Le Guernic 1990]
- Simulink [Ciolfi et al., 1990s]
- Statecharts [Harel 1987]
My Agenda

I will show a particular approach to the design of concurrent and distributed time-sensitive systems that is an actor-oriented component technology.

The approach is called PTIDES (pronounced “tides”), for Programming Temporally Integrated Distributed Embedded Systems.
Our Approach is based on Discrete Events (DE)

- Concurrent actors
- Exchange time-stamped messages (“events”)

A correct execution is one where every actor reacts to input events in time-stamp order.

Time stamps are in “model time,” which typically bears no relationship to “real time” (wall-clock time). We use superdense time for the time stamps.
Example DE Model (in Ptolemy II)

**DE Director** specifies that this will be a DE model.
Example DE Model

Model of regularly spaced events (e.g., a clock signal).
Example DE Model

Model of irregularly spaced events (e.g., a failure event).
Example DE Model

Model of a subsystem that changes modes at random (event-triggered) times
Example DE Model

Model of an observer subsystem
Example DE Model

Events on the two input streams must be seen in time stamp order.

Note that DE MoCs have considerable subtleties when it comes to simultaneous events and events that prevent time from progressing (Zeno conditions).

Super Dense Time Enables Better Conjunction of Computation and Physical Processes

Values $V$:

Initial segment $I \subseteq \mathbb{R}_+ \times \mathbb{N}$ where the signal is defined

Absent: $s(\tau) = \varepsilon$ for almost all $\tau \in I$. 

Lee, Berkeley 15
This is a Component Technology

Model of a subsystem given as an imperative program.
This is a Component Technology

Model of a subsystem given as a state machine.
This is a Component Technology

Model of a subsystem given as a modal model.

More types of components:
• Modal models
• Functional expressions.
• Submodels in DE
• Submodels in other MoCs
Using DE Semantics in Distributed Real-Time Systems

- DE is usually a simulation technology.
- Distributing DE is done for acceleration.
- Hardware design languages (e.g. VHDL) use DE where time stamps are literally interpreted as real time, or abstractly as ticks of a physical clock.

- We are using DE for distributed real-time software, binding time stamps to real time only where necessary.
- *PTIDES*: Programming Temporally Integrated Distributed Embedded Systems
PTIDES: Programming Temporally Integrated Distributed Embedded Systems

Distributed execution under discrete-event semantics, with “model time” and “real time” bound at sensors and actuators.
PTIDES: Programming Temporally Integrated Distributed Embedded Systems

PTIDES uses static causality analysis to determine when events can be safely processed (preserving DE semantics).

Assume bounded sensor delay $s$

Assume bounded network delay $d$

Assume bounded clock error $e$

An earliest event with time stamp $t$ here with time stamp $t$ can be safely merged when real time exceeds $t + s + d + e - d^2$
Schedulability analysis incorporates computation times to determine whether we can guarantee that deadlines are met.
PTIDES: Programming Temporally Integrated Distributed Embedded Systems

... and being explicit about time delays means that we can analyze control system dynamics...

Actuator may process the event at the time received or wait until real-time matches the time stamp. The latter yields determinate latencies.

Feedback through the physical world
Experimental Setup

- **Ptides Model**
  - Ptolemy II Ptides domain

- **Code Generator**
  - Ptolemy II Discrete-event, Continuous, and Wireless domains

- **Mixed Simulator**
  - Luminary Micro 8962

- **HW in the Loop Simulator**
  - Luminary Micro 8962

- **Software Component Library**
  - IEEE 1588 Network time protocol

- **HW Platform**
  - Luminary Micro 8962

Analysis

- **Schedulability Analysis**
  - Experimental Setup

- **Causality Analysis**
  - HW Platform

- **Program Analysis**
  - Component Library

- **Ptides Model**
  - Code

- **PtidyOS Code**

- **Ptolemy II Ptides domain**

- **Ptolemy II Discrete-event, Continuous, and Wireless domains**

- **Network Model**
  - Luminary Micro 8962

- **Plant Model**

- **Code**

- **HW Platform**