Synthesis of Distributed Real-Time Embedded Software

Edward A. Lee
Robert S. Pepper Distinguished Professor
UC Berkeley

Keynote talk

Electronic System Level Synthesis Conference
June 5-6, 2011
San Diego, California, USA

Key Collaborators:
• Steven Edwards
• Sungjun Kim
• Isaac Liu
• Slobodan Matic
• Jan Reinke
• Sanjit Seshia
• Mike Zimmer
• Jia Zou
Cyber-Physical Systems (CPS):
Orchestrating networked computational resources with physical systems

Automotive
- E-Corner, Siemens
- Daimler-Chrysler

Military systems:

Power generation and distribution
- Courtesy of General Electric

Transportation (Air traffic control at SFO)

Avionics

Instrumentation (Soleil Synchrotron)

Building Systems

Telecommunications

Factory automation
Printing Press Example

- Application aspects
  - local (control)
  - distributed (coordination)
  - global (modes)
- Open standards (Ethernet)
  - Synchronous, Time-Triggered
  - IEEE 1588 time-sync protocol
- High-speed, high precision
  - Speed: 1 inch/ms
  - Precision: 0.01 inch
    -> Time accuracy: 10us

Bosch-Rexroth

Orchestrated networked resources built with sound design principles on suitable abstractions

DETERMINISM TIMED SEMANTICS
Claim

For CPS, *programs do not today adequately specify behavior.*
Problems that complicate analysis of system behavior:

- Sensors may be locked out for an indeterminate amount of time.
- Variability of execution times affects results (not just WCET).
- Messages from different sources interleave nondeterministically.
- Platforms’ measurements of time differ.
- A fault in a remote component may disrupt a critical local activity.
- A fault in a remote component may go undetected for a long time.
- Interrupt-driven I/O disrupts timing.

Etc…
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.
Building a DE Model (in Ptolemy II)

DE Director specifies that this will be a DE model
Building a DE Model (in Ptolemy II)

Model of regularly spaced events (e.g., a clock signal).
Building a DE Model (in Ptolemy II)

Model of irregularly spaced events (e.g., a failure event).
Building a DE Model (in Ptolemy II)

Model of a subsystem that changes modes at random (event-triggered) times
Building a DE Model (in Ptolemy II)

Model of an observer subsystem
Building a DE Model (in Ptolemy II)

Events on the two input streams must be seen in time stamp order.
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 traditionally 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: First step: Time-stamped messages.
Ptides: Second step: Network time synchronization

GPS, NTP, IEEE 1588, time-triggered busses, etc., all provide some form of common time base. These are becoming fairly common.
Time-Aware Networking Technology Facilitates Network Time Synchronization

- Frequency locking
  - Enables integrating circuit-switched services on packet-switched networks
  - Can deliver performance independent of network loading.

- Time synchronization
  - E.g., *IEEE 1588* standard set in 2002.
  - Synchronized time-of-day across a network.
Precision Time Protocol (PTP) Standardized for Ethernet

This may become routine!
With this PHY, clocks on a LAN agree on the current time of day to within 8ns, far more precise than older techniques like NTP.
An Extreme Example: The Large Hadron Collider

The WhiteRabbit project at CERN is synchronizing the clocks of computers 10 km apart to within about 80 psec using a combination of IEEE 1588 PTP and synchronous ethernet.
Ptides: Third step: 
Bind time stamps to real time at sensors and actuators

Input time stamps are ≥ real time
Output time stamps are ≤ real time
Clock synchronization gives global meaning to time stamps
Messages are processed in time-stamp order.
Ptides: Fourth step: Specify latencies in the model

Global latencies between sensors and actuators become controllable, which enables analysis of system dynamics.

Model includes manipulations of time stamps, which control latencies between sensors and actors.

Actuators may be designed to interpret input time stamps as the time at which to take action.

Feedback through the physical world
Ptides: Fifth step
Safe-to-process analysis (ensures determinacy)

Safe-to-process analysis guarantees that the generated code obeys time-stamp semantics (events are processed in time-stamp order), given some assumptions.

Assume bounded sensor delay $s$
Assume bounded network delay $d$
Assume bounded clock error $e$
Application specification of latency $d_2$

An earliest event with time stamp $t$ here can be safely merged when real time exceeds $t + s + d + e - d_2$
Delivering Bounded Network Delay
Domain-Specific Networks capable of Bounded Delay

- **WorldFIP** (Factory Instrumentation Protocol)
  - Created in France, 1980s, used in train systems
- **CAN**: Controller Area Network
  - Created by Bosch, 1980s/90s, ISO standard
- Various **ethernet** variants
  - PROFInet, EtherCAT, Powerlink, …
- **TTP/C**: Time-Triggered Protocol
  - Created around 1990, TU Vienna, supported by TTTech
- **MOST**: Media Oriented Systems Transport
  - Created by a consortium of automotive & electronics companies
  - Under active development today
- **FlexRay**: Time triggered bus for automotive applications
  - Created by a consortium of automotive & electronics companies
  - Under active development today
Ptides Schedulability Analysis
Determine whether deadlines can be met

Schedulability analysis incorporates computation times to determine whether we can guarantee that deadlines are met.

Deadline for delivery of event with time stamp $t$ here is $t - c_3 - d_2$

Assume bounded computation time $c_1$

Assume bounded computation time $c_2$

Deadline for delivery here is $t$

Assume bounded computation time $c_3$
PtidyOS: A lightweight microkernel supporting Ptides semantics

Current prototype runs on a COTS Arm platform (Luminary Micro) with rudimentary support for IEEE 1588 network time synchronization. Occupies about 16 kbytes of memory.

Currently porting to Renesas and PRET platforms.

An interesting property of PtidyOS is that despite being highly concurrent, preemptive, and EDF-based, it does not require threads. A single stack is sufficient!
Workflow Structure

Analysis

Schedulability Analysis

Causality Analysis

Program Analysis

Ptides Model

Code Generator

Software Component Library

Mixed Simulator

Plant Model

Network Model

Ptides Model

Code

PtidyOS

HW Platform

HW in the Loop Simulator

Luminary Micro 8962

IEEE 1588 Network time protocol

Ptolemy II Ptides domain

Ptolemy II Discrete-event, Continuous, and Wireless domains

Lee, Berkeley 29
A Test Case for PtidyOS

This device, designed by Jeff Jensen, mixes periodic, quasi-periodic, and sporadic real-time events.

Tunneling Ball Device
– sense ball
– track disk
– adjust trajectory
Tunneling Ball Device in Action
Tunneling Ball Device

**Mixed event sequences**

- **Periodic Events**
- **Quasi Periodic Events**
- **Sporadic Events**
Ptides Project Status

- Seed funding from ARL got the project going.
- Ongoing NSF effort (CPS Medium)
  - Sanjit Seshia focused on WCET & schedulability analysis
  - Ptolemy II-based simulator supports multiform clocks
  - PtidyOS being prepped for open-source release
Ptides Publications

Conclusions

Today, timing behavior is a property only of realizations of software systems.

Tomorrow, timing behavior will be a semantic property of programs and models.

Raffaello Sanzio da Urbino – The Athens School