MODEL-BASED/
PLATFORM-BASED/
ARCHITECTURE-DRIVEN
DESIGN
OF CYBER-PHYSICAL
SYSTEMS

Patricia Derler
University of California, Berkeley
A PTOLEMY MODEL
FOCUS IN THE PTOLEMY PROJECT

• Modeling, simulation, and design of concurrent, real-time, embedded systems

• Executable models rather than descriptive models

• Semantics matters (more than syntax)

• Useful semantics imply constraints on designers

• Heterogeneity may be better than generality

Freedom from choice!
RECENT WORK: PLATFORM-BASED DESIGN IN PTOLEMY II

Executable Model of Functionality

Add architectural artifacts to the functionality model
DISTRIBUTED REAL-TIME SYSTEMS

Multiple computers, comprising of sensors and actuators, connected on a network that act and react on events to meet timing constraints.

Automotive

Telecommunications

Building Systems

Factory automation

Power generation and distribution

Military Systems

Avionics

Transportation

(Air traffic control at SFO)

Courtesy of Kuka Robotics Corp.

Instrumentation

(Soleil Synchrotron)

Courier of General Electric

Courtesy of Doug Schmidt
MODELING DISTRIBUTED REAL-TIME SYSTEMS
MODELING DISTRIBUTED SYSTEMS

Director mediates between actors
Difficult to maintain different notions of time

One hierarchy level is not enough
Hierarchies:
- Opaque composite actors
- Embedded directors maintain time
THE TIME CHALLENGE

Distributed platforms have different notions of time

Platform clocks drift

Platform clocks drift at varying rates

Multiple clocks on a single platform: SW/HW clocks
MODELING DISTRIBUTED SYSTEMS

Top level: Oracle time

Platform time is defined with respect to oracle time

Platform1 time = $f_1(t_0)$

Platform3 time = $f_3(t_0)$

Platform2 time = $f_2(t_0)$
CLOCK SYNCHRONIZATION

Master

Slave

adjust clock rate:

adjust clock value:
Distributed platforms communicate via networks. Networks have latencies, e.g., CAN Bus, TTEthernet.
Physical connections vs. Logical connections

Logical connections are lost
Aspect-oriented modeling

Quantity managers [Balarin03] and schedulers to simulate network latency

MODELING EXECUTION TIME

.execution time

Network fabric

Platform 1

Sensor1 → Computation1

Platform 2

Sensor2 → Computation2

trigger

physical interface

Local Event Source

Clock

Physical plant

Computation3 → Merge

Merge → Actuator1

physical interface
DISTRIBUTED DISCRETE-EVENT MODELS

DE as a application specification language, semantic basis for obtaining determinism in distributed real-time systems.

Event
(Time stamp \( t' \), Value \( v' \))

Event
(Time stamp \( t \), Value \( v \))

Director

Platform1

Platform2

Platform3

Physical Plant

Computation3

Merge

Actuator1

Director

Computation4
DISTRIBUTED DISCRETE-EVENT MODELS

- Platform 1 time
- Logical time

- Platform 2 time
- Logical time

- Platform 3 time
- Logical time

Logical time describe the execution semantics

New time line: logical time

Oracle time

Platform time

Logical time
PTIDES: AN APPLICATION

Programming temporally integrated distributed event systems [Zhao07]

- Discrete event model for execution
- Relates logical time to platform time at sensors, actuators and network interfaces
- Requires bounded error between platform clocks: Relies on clock synchronization
- Causally related events are processed in time-stamped order

A PTIDES Model

- **Time stamp = Platform time**
- **Platform time progresses**
- **Send time stamp and value and over network**
- **Response time ≤ Logical time delay**
- **Increase time stamp**
- **Time stamp ≤ Platform time**

**Oracle time**

**Platform time**

**Logical time**

**Time stamp ≥ Platform time**

**Sensor delays, dynamic network latencies, clock synchronization error**

**Physical interface**

**Physical plant**
EXPLICIT SPECIFICATION OF TIME

response time = d4 + d2

classification of end-to-end sensor to actuator delays
**EXPLICIT SPECIFICATION OF TIME**

- Behavior independent of distribution

\[ \text{response time} = d_4 + d_2 \]
PARALLEL EXECUTION
PARALLEL EXECUTION

Safe-to-Process Analysis

Event = (timestamp, event)

Can an event at this port arrive with a smaller timestamp later in the execution? Evaluate other events, Sensor input patterns, delays, dependencies, actor topologies, scheduling strategy, ...
PARALLEL EXECUTION

Single Core
PARALLEL EXECUTION

Multi Core
PTIDES WORKFLOW

- **Analysis**
  - Schedulability Analysis
  - Causality Analysis
  - Program Analysis

- **Ptides Model**
  - Code Generator
  - Software Component Library
  - PtidyOS
  - HW Platform
  - HW in the Loop Simulator

- **Mixed Simulator**
  - Plant Model
  - Network Model

- **Graphs**
  - Reserve Velocity (red), Target Velocity (green) and Tracking Error
  - Time in seconds
  - Velocity m/s
  - Contact (red), Top Dead Center (green), Cut (blue) and Arm (black)
  - Time in seconds

- **Code**
  - Program Analysis

- **Images**
  - Circuit diagram
  - Microcontroller