Motivating Hierarchical Run-Time Models for Measurement and Control Systems

Jie Liu†, Stan Jefferson‡, and Edward A. Lee†

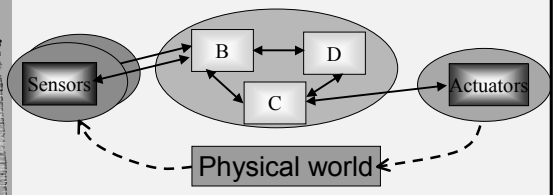
†Ptolemy Group
EECS, UC Berkeley
{liuj, eal}@eecs.berkeley.edu

‡Agilent Laboratories
Agilent Technologies
stan_jefferson@labs.agilent.com

Ptolemy Miniconference
Berkeley, CA, March 22-23, 2001

Measurement and Control Systems

- Distributed real-time systems
  - Real-time
  - Concurrent
  - Interacting with the physical world



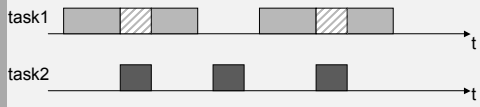
What characterizes a run-time model?

- Messages Semantics
  - event vs. state
- Message Acquisition Styles
  - push vs. pull
- Dataflow and Control flow
  - Data dependencies
  - Scheduling
  - Mode switching
- Notion of Time

Examples of Run-Time Models

- Priority-driven multitasking
- Time-triggered architectures
- State machines
- Dataflow models
- Publish and subscribe
- Time-based event-driven

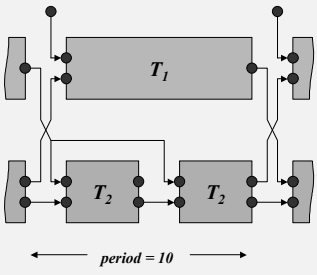
Priority-driven multitasking (RTOS)



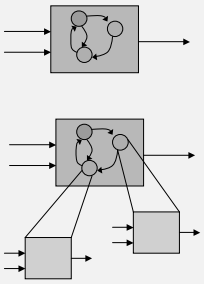
- Tasks are finite computation
- Priorities are assigned to tasks to share resources
- Based on RMA or EDF scheduling theories
- Assume tasks are arbitrarily preemptable
- Hard to distribute

Time-Triggered Architectures (TTA)

- Periodically scheduled tasks
- Time-triggered computation and communication
- Precisely defined data exchange time
- Easily verifiable timing properties
- Inflexible handling of sporadic events

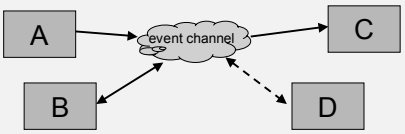


State Machines



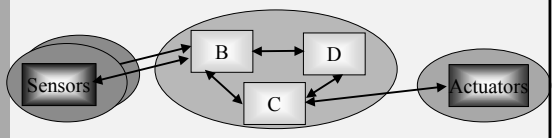
- Within an atomic component
  - Well-defined state
  - Precise reactions
- Coordinate Concurrent Models
  - Modal behavior
  - Operation sequences

Publish and Subscribe



- Primarily a distributed model for three tier communication
- Event channel mediates senders and receivers
- Good for modeling peers that come and go
- Event dispatching can also be prioritized

Time-Based Event Driven

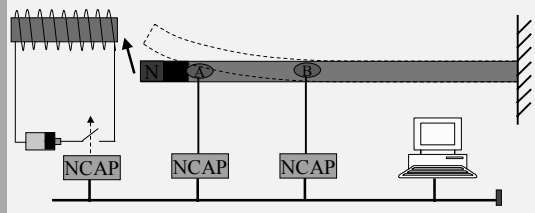


- Global synchronized time on distributed nodes
- Sensors time tag their readings
- Actuators set timers to produce output at desired time
- Computers process events as fast as possible

Hierarchical Run-Time Models

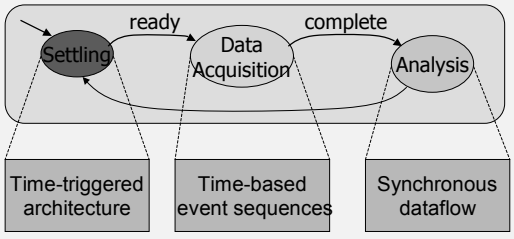
- Single flat layer of traditional real-time operating system is rigid and fragile
- Hierarchical run-time based on Ptolemy II component architecture and models of computation
  - Component architecture
    - Well-defined communication points
    - Polymorphic execution interface
    - Separation of data transfer and flow of control
  - Models of computation
    - Hierarchical composition
    - Domain-specific receivers
    - Domain-specific execution order
    - Composable atomic execution

Example: A Data Acquisition & Analysis System

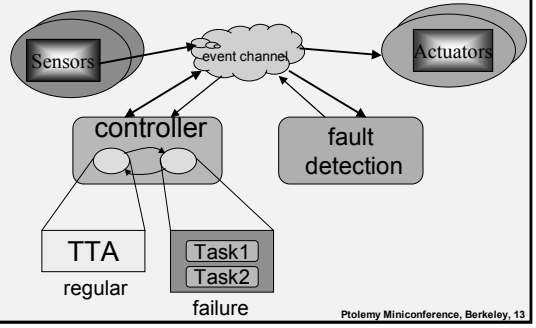


- Time-based and event-triggered sequential operations
- Time-synced sensor data acquisition
- Composition of timed and untimed models

Hierarchical Run-Time Models



Example: Fault-Tolerant Control



Conclusion – Run-Time & Design-Time Environments

- **Differences**
  - **There are certain models that are design-time only**
  - **Design-time environments emphasize on understandability, syntax and semantics checking, and component reuse**
  - **Run-time environments emphasize on physical interface, performance, and footprint.**
- **Synergy**
  - **component-based design → component-based execution**
  - **Simplifies code generation**
  - **Platform dependent implementation**