CONSORTIUM

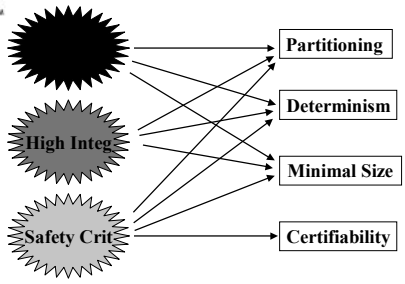
Model for Supporting High Integrity and Fault Tolerance

Brian Dobbing, Aonix Europe Ltd
Chief Technical Consultant
November 1999

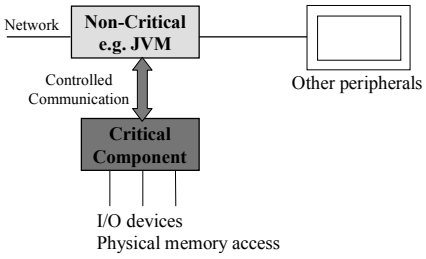
Background

- Real-Time Core Specification is Released
- High Integrity Profile
  - A subset of APIs in the RT-Core
  - Address requirements of
    - Fault Tolerance
    - High Integrity and Safety Critical systems
  - Work in progress, led by Aonix
    - Draft circulated for internal J-Consortium review
    - CALL FOR PARTICIPATION e.g. from Telecoms

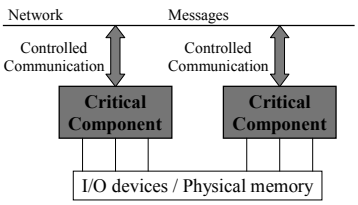
Characteristics



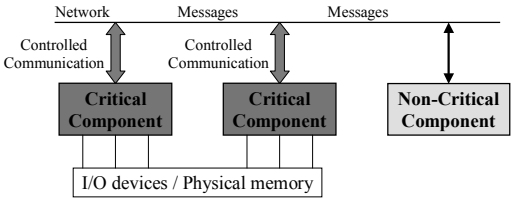
Partitioning - single processor



Partitioning - hot standby

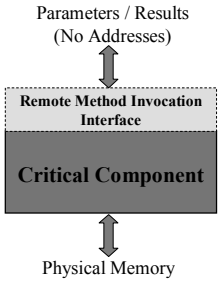


Partitioning - distribution



- Support for Distributed Systems is being addressed by "Real-Time and Embedded Distributed Middleware Profile"

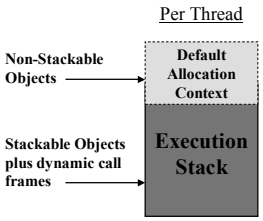
Partitioning - interfaces



Partition Construction

- Static Linking only
  - Need to verify statically every byte loaded
- Native code execution only
  - Temporal constraints prohibit interpreters
- "Control" of the processor
  - HI partition will (normally) contain main()
  - Low-level interaction with underlying board
    - Intended to be user-configurable

Memory Management (1)



No garbage collector!

Memory Management (2)

- Allocation context re-used for each run()
  - Useful for Periodic, Sporadic, Interrupt tasks
- Special allocation context supported
  - Programmatic reclamation for Ongoing tasks
- Allocation context must be non-fragmenting

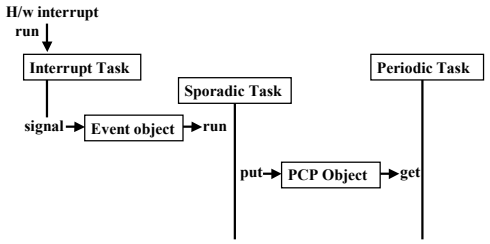
Concurrency

- Periodic tasks (threads)
  - Scheduled by the runtime system
- Sporadic tasks (threads)
  - Additional to RT Core, run by signalling Events
- Interrupt "tasks" (handlers)
  - Triggered by hardware / software interrupt
- Ongoing tasks (not run automatically)
  - Often background tasks in an infinite loop

Task Interaction

- Remove asynchronous task-to-task actions
  - Stop, Interrupt, Wakeup, Suspend / Resume
  - Important for replica determinism
- PCP used for Synchronized methods
  - Mutual exclusive access to Shared Resources
  - Tight bound on the blocking time (no queues)
  - Used in preference to mutexes, semaphores
- Atomic regions used for interrupt handlers
  - Can signal Event object to run Sporadic task

Task Interaction Example



Scheduling

- Priorities as in the RT-Core
  - Interrupt range + non-interrupt range
- Normal FIFO_Within_Priorities scheduling
- Task inherits ceiling priority when in PCP
- Each task has :
  - Base priority (at creation time)
  - Active priority (for scheduling)
    - Higher of base and any inherited priority

Sequential Execution

- Need to be able to support :
  - Access to physical memory addresses
    - for example RT Core IO class and Device I/O Registry
  - Exception handling
    - Needed for failure recovery (roll back state)
  - Finally clauses
    - Tidy up when failure occurs
- Must exclude any non-determinism
  - Needed for replica consistency in fault tolerance

Fault Tolerance (1)

- Hardware-related considerations
  - Keep standby replicas as identical as possible
    - Eliminate constructs with race conditions
    - Predictable execution even if clocks vary slightly
  - Allow access to physical addresses
    - e.g fast DMA to compare / update state in replica
  - No addresses exported from HI partition
    - Simplifies switching to replica on failure

Fault Tolerance (2)

- Software-related considerations
  - Checkpoint objects at specific points
    - Save state in durable (persistent) storage
    - Allow access to physical addresses for this storage
  - Detect and recover from data errors
    - Exception catching
    - Roll back to saved state
  - Tidy up in normal and exception case
    - Finally clauses

Summary (1)

- The HI Profile supports :
  - Partitioning
    - Controlled access to code and data
  - Determinism in sequential and concurrency
  - Small footprint runtime system
- Proposed features :
  - Simplified memory management scheme
    - Including access to physical memory addresses
  - Simplified threads kernel
  - Full exception handling and “finally” clauses

Summary (2)

- The HI Profile will be :
  - Compatible with the Real-Time Core definition
  - Compatible with the Distributed Middleware profile
- A full solution for the entire range of software requirements

Needs YOUR review comments & participation