An Evolution of QoS Context Propagation in Event Mediated Avionics Software Architectures

Christopher D. Gill and Joseph W. Hoffert
{cdgill,joeh}@cs.wustl.edu
Department of Computer Science
Washington University, St. Louis, MO

David C. Sharp and Patrick H. Goertzen
{david.c.sharp, patrick.h.goertzen}@boeing.com
The Boeing Company, St. Louis, MO

20th IEEE/AIAA DASC
Wednesday, October 17, 2001

Work Supported by Boeing, AFRL
Motivation

• Avionics mission computing use-cases have differing levels of QoS requirements
  – *E.g.*, situational awareness vs. actuator control
• QoS management capabilities trade off with temporal and spatial “overhead” factors
• A given application should receive “just enough QoS management”
  – *i.e.*, sufficient control with minimal overhead
• Goal: an architecture and implementation framework that scales across use-cases
  – Supports re-use of QoS management capabilities
  – Can be customized to specific use-cases
  – Application only “pays” for QoS management it uses
Motivation, Continued

• QoS context is propagated for end-to-end management:
  – In-band, i.e., in the run-time path of the managed info
  – Or, out-of-band, i.e., through another path

• Thread model for QoS context propagation well known
  – E.g., in-band pass a timeout parameter to a function call
  – E.g., out-of-band set the priority of a thread
  – E.g., distributable thread scheduling in DSRTCORBA2

• We consider implications for an event-mediated model
  – I.e., in-band with the event + several out-of-band techniques
  – Event, rather than thread is primary concurrency abstraction
  – Useful for highly asynchronous distributed applications
    • E.g., situational awareness in avionics mission computing
## Kinds of Propagation Models

<table>
<thead>
<tr>
<th>Model</th>
<th>Example Application</th>
<th>Application Characteristics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Statically Scheduled</td>
<td>Mission Computing Pilot-Vehicle Interface</td>
<td>Total utilization is static (within known modes)</td>
</tr>
<tr>
<td>Hybrid Static/Dynamic</td>
<td>System Health Management</td>
<td>Total utilization is intentionally or inherently dynamic</td>
</tr>
<tr>
<td>Adaptively Scheduled</td>
<td>Synthetic Aperture Radar (SAR) Mapping</td>
<td>Modes are defined adaptively for better overall resource control</td>
</tr>
<tr>
<td>Deterministic/Statistical</td>
<td>Enroute Mission Re-planning</td>
<td>Inherent variability of utilization by some or all operations</td>
</tr>
</tbody>
</table>
Original Static Priority Model

- Event pushed to dispatcher
- Event carries handle to its static QoS info (prio)
  - Low space overhead
- Dispatcher calls scheduler w/ handle, gets prio lane
  - Low time overhead
- Dispatcher enqueues event in (pre-configured) lane
- Lane thread priorities and FIFO queues enforce scheduling strategy
  - i.e., RMS
Hybrid Static/Dynamic Model

- Event carries static handle + dynamic QoS info
  - *E.g.*, deadline, WCET
  - More space overhead
- Dispatcher looks up lane
  - Same as for static model
- Dispatcher enqueues event in (possibly dynamic) lane
  - *E.g.*, time to deadline
  - More time overhead
- Lane thread priorities and queue disciplines enforce scheduling strategy
  - *I.e.*, RMS, MUF, RMS+LLF
Adaptively Scheduled Model

In Band
- Push to Dispatcher
- Lookup in Scheduler
- Enqueue in lane
- Adapter intercepts Component push
- Push to Component

Out of Band
- Report to Monitor
- QoS Manager queries Monitor (separate event)
- May trigger update in Scheduler
Hybrid Deterministic/Statistical Model

In Band
- TNA Scheduler pushes to Dispatcher
- Lookup in Scheduler prior & budget
- Enqueues in lane
- Push to special TNA event consumer
- TNA consumer executes task

Out of Band
- Budget update step
  *E.g., in SRMS*
Further Extensibility (All Models)

- Arbitrary info can be passed
- Event header payload is a CORBA::Octet sequence
- Optimized in TAO
  - uses template specialization
  - efficient single-copy semantics
- E.g., C++ pointer if local
- E.g., CORBA IOR if remote
- E.g., using Command Pattern for local or remote invocation
# Synopsis of QoS Context Propagation

<table>
<thead>
<tr>
<th>Model</th>
<th>Additional Context</th>
<th>QoS Evolution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Statically Scheduled</td>
<td>handle</td>
<td>Priorities set off-line prior to run-time</td>
</tr>
<tr>
<td>Hybrid Static/Dynamic</td>
<td>deadline worst case execution</td>
<td>Run-time scheduling may be dynamic <em>(e.g., non-critical operations)</em></td>
</tr>
<tr>
<td>Adaptively Scheduled</td>
<td>deadline statistics operation rate &amp; priority</td>
<td>Optimize adaptively by rescheduling – manage variable environments</td>
</tr>
<tr>
<td>Deterministic/Statistical</td>
<td>budgets debits (on completion) refresh (at super-period)</td>
<td>Assess/enforce weaker assurances w/ inherent variability in application</td>
</tr>
</tbody>
</table>

key: *in-band propagation* vs. *out-of-band propagation*
Concluding Remarks

- Common Architecture Across Use Cases
  - Extends capabilities for dealing with QoS variation
    - Dynamic and hybrid static/dynamic scheduling
    - Adaptive monitoring and re-scheduling
    - Handling a range of predictability in execution times
  - “Weight” of mechanisms customizable to specific use-case

- Future Work
  - Effects of explicitly vs. implicitly known variability
  - Adaptation among scheduling heuristics
  - Fine grain examination of use case factors / mechanisms
    - E.g., cancellation strategy / consecutive vs. independent route legs

- Thanks: Work Supported By
  - AFRL and OSJTF (ASTD, ASFD, WSOA)
  - DARPA ITO (Quorum, NEST)
  - The Boeing Company (IRAD)