

RT Aggregated Systems

untapped potential, unsolved problems

A proposed Open Group Challenge

Open Group Cannes Meeting Presentation

Dock Allen

Open Group RT/QoS Forum Liaison

OMG RTESS Chair

The MITRE Corporation

Dock@MITRE.org 781 271 8216

Outline

- Aggregated Systems
- Software Enclaves
- Real-time
- Available technology
- Future Technologies
- Challenge

Aggregated Systems

- Definition
 - Systems of systems, integration of independently developed systems
- Characteristics
 - Multiple infrastructures with different ways of managing QoS/timeliness
 - Unpredictable network loads, which can increase significantly due to interference and transient errors
 - Difficult / impossible to get a “God’s eye view”

Software Enclaves

- Multiple software enclaves exist
 - Data-base programming (SQL, etc)
 - Procedural programming (C++, Java, etc)
 - Parallel programming (C, signal processing)
 - Safety critical systems development (Ada)
- Enclaves have their own preferred languages, infrastructures, processes, tools, and vendor communities
- There is remarkable little overlap
- Aggregated systems often span enclaves

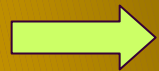
Res Temporalis



- time critical data and operations - usefulness degrades over time, consequences of missing deadlines can be serious (*hard and soft Real-time*)
 - e.g. Stock trades, targeting applications



- temporal responsiveness of interactions or *interactive real-time* (system-human and system-system)
 - Online human interactions, computer-computer interactions, where the other side will “time out” is response is too slow



- accurate temporal pacing of streams of data (with acceptable temporal delay/latency)
 - Streaming video

Res Temporalis (other)

- time sensitive (perishable) data and operations
 - Stock prices, moving target imagery
- Correct temporal ordering of data and requests
- temporal coherence of data from different sources
 - image fusion (usually managed by the application)
- conditional workflow execution based on temporal relationships

Timeliness Concerns

- Timeliness of data / operations is handled differently in each enclave (and sometimes within enclaves)
- How do we build Real-time systems that span enclaves or different infrastructures within an enclave?
- How do we know that these systems will work?

Some Available Technologies

- Splice - real-time for European military systems with a data-base flavor
- RT CORBA - state of the practice for RT middle-ware
- RT Java - a similar paradigm to RT CORBA, support is emerging
- RT Message Oriented Middleware (e.g. RTI)
- RT extensions to UML
- Transport options including IPV6 with priority support, DiffServ, MPLS, U4EA, InfiniBand, etc

Interesting Research

- DARPA QORUM/QOIN adaptive resource management
- MITRE research on application QoS
- MITRE research on RT Web Services
- Brandeis temporal markup work (TenseML)
- ...

Emerging Technologies

- Web Services Environments
 - Some vendors support integration of diverse infrastructures (super structures)
 - Has the potential to integrate the information-centric and procedural enclaves
 - Has no QoS or timeliness support -- yet
- OMG and Open Wings RT Data Distribution Service
- Distributed Real-time Java
- Network QoS and Service Level Agreements

What can we do?

- Issue a vendor challenge
 - Issued by the Open Group
 - Supported by the OMG
 - Brings users, researchers, vendors together
 - prototype partial solutions
 - identify gaps

What we need in a scenario

- Represent aggregated system
 - multiple enclaves (at least 2)
 - database --parallel
 - procedural --safety critical
 - heterogeneous infrastructures
- Dependable end-to-end timeliness required for some of the applications
- Should be performance challenging, but achievable
- Should involve enterprise integration
 - requires a QoS framework
- individual apps should be doable
- should include non-R/T apps
- multi-lingual not required and not excluded
- some dynamic work loads

- Separate out analytical versus demo versus other types of evidence
- define level and type of dependability and the measurement approach
- need to show different loads on different resources (large variations) to show that solution is robust
- solutions need to be robust to the independent evolution of application requirements
- remember policy driven QoS (multi-dimensional)

Discussion

- What are the other requirements for scenarios (your turn)
