Using a Real-time, QoS-based ORB to Intelligently Manage Communications Bandwidth in a Multi-Protocol Environment

Bill Beckwith
Objective Interface Systems, Inc.
+1 703 295 6500
bill.beckwith@ois.com
http://www.ois.com
OMG Real-time and Embedded Workshop
July 2002

The Nature of CORBA

- ORBs abstract away network semantics
  - Application design is much less constrained by the communications infrastructure
  - *But* the application developer has little direct control over the communication paths and parameters used
- Thus, ORBs make it difficult to
  - Control communication channels
  - Much less manage communications bandwidth
  - Much less schedule the communications channels

The Nature of Real-time CORBA

- Real-time CORBA provides abstracts for timeliness
  - Allows applications to schedule distributed schedule processor workload
  - Real-time CORBA spec provides ProtocolProperties for controlling communication resources on a per ORB basis
  - *But*, real-time ORBs do not typically allow for coordination of processor resources with communication resources
- Thus, Real-time CORBA ORBs still make it difficult to
  - Manage communication bandwidth on a connection-by-connection basis
  - Granular control over scheduling of communication resources

What is QoS?

- QoS = Quality of Service
- QoS means different things to different people
- QoS definition for this presentation
  - The parameters offered by a communications transport for affecting the characteristics of one communication channel vs. another
  - *Not* a scheduling parameter (see Dynamic Scheduling)

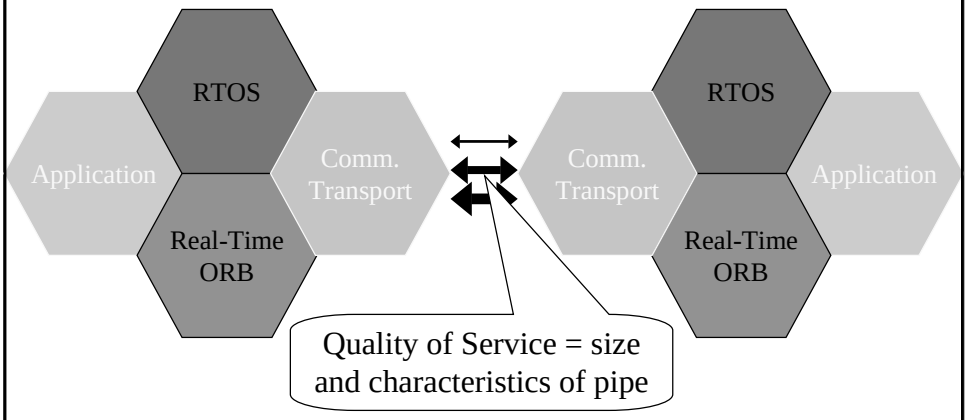
Why QoS?

- Facilitates transports that can
  - maintain priority
  - distribute bandwidth
  - guarantee jitter
  - bound latency
  - etc.
- A needed general abstraction of what transports offer

Most ORBs Ignore QoS

- Middleware and ORBs have traditionally ignored transport QoS
- Middleware can't tell transport to favor one connection over another
- Middleware must live with either
  - Bad: Default settings or
  - *Worse:* semantics of TCP layered on top of native transport

Quality of Service in a Real-time CORBA Application



Processor Scheduling

- Most real-time theory ignores the unsolvable problem
  - Scheduling multi-processor, multi-node, multi-transport distributed systems
  - Instead focuses on simpler scheduling of **one** processor in **one** system
- The harder problem is natural to the real-world problems
  - Exists in many real-time systems
  - Is not going away

End-to-end Latency and Jitter

- Crucial to identify and isolate the sources of latency and jitter
  - Key to understanding the benefits of QoS
- Contributors to latency and jitter
  - Application
  - Real-time ORB
  - Replaceable transport plug-in
  - Communications transport (protocol stack and media)*
  - Operating system (thread dispatching, interrupts, …)
  - Higher priority activities (interrupts, …)

** Frequently the largest contributor in my experience*

Application Latency and Jitter

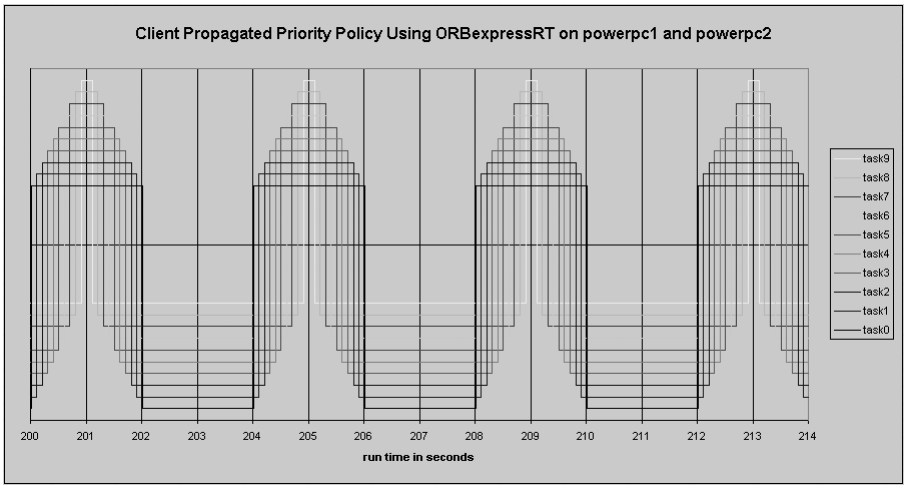
- Mostly within the control of the application engineer
- ORB should facilitate application's control of scheduling requirements (i.e. Dynamic Scheduling for CORBA)
- ORB should aid developer in avoiding priority inversions and priority deadlock

Real-time ORB Latency and Jitter

- A virtue or vice of the ORB implementation
- Depends on the quality of ORB implementation
- Is well within the control of the real-time ORB implementer
- A well implemented real-time ORB should add very little or no jitter
- Real-time ORBs exist that add very little latency
- Real-time ORBs exist without priority inversions

Real-Time ORB Priority Testing

Boeing Second Phase Test Results – Nov 20, 2000



Replaceable Transport Plug-in Latency and Jitter

- Also may be developed by application engineer
- Can introduce additional latency and jitter if poorly designed
- A properly engineered transport *shouldn't* introduce jitter

Communications Transport Latency and Jitter

- Frequently *the* significant source of latency and jitter
- More advanced transports offer hope
  - Hope is spelled "Q" "o" "S"
  - Some are bandwidth oriented
  - Others are priority oriented

Operating System Latency and Jitter

- Easy to test for, hard to isolate
- Caused by:
  - Poor scheduling/dispatching
  - Badly designed interrupt handlers
  - Non-real-time protocol stacks
  - Bad algorithms in support libraries
    - Watch out for printf()!
  - Poor priority management in support libraries
    - No priority inheritance in O/S mutexes
    - Running protocol stacks at lower priorities
- Only use RTOSes for real-time systems

Jitter Caused by High Priority or Other Activities

- May or may not be under application developers control
- Where possible, remove spurious interrupts
  - If your invocations across IEEE 1394 are showing large jitter:
    Unplug the Ethernet cable from your board!
- Bus contention by I/O devices, etc.

Various Communications Transports

- Ethernet
  - Random hardware interrupts
  - Variable workload caused by each interrupt
  - Hubs add less latency, more jitter
  - Switches add latency, less jitter
  - No QoS with standard Ethernet
  - Various technologies for switched Ethernet QoS

Various Communications Transports (cont.)

- TCP/IP
  - Fine with Ethernet if you have no low latency, bounded latency or jitter requirements ;-)
  - Needlessly duplicates reliability that may be available as a QoS parameter in a lower level transport (e.g. ATM virtual circuits)
  - Many protocol stacks turn Ethernet's random interrupts into random workloads

Various Communications Transports (cont.)

- ATM
  - Random hardware interrupts
  - Variable workload caused by each interrupt
  - Complex set of QoS parameters allow some throttling of data flow
- IEEE 1394/FireWire
  - Periodic interrupts
  - Variable workload caused by each interrupt
  - QoS is isochronous bandwidth allocation

Various Communications Transports (cont.)

- Reflective memory
  - *No* hardware interrupts
  - Contention for media is bounded
  - Highly predictable
  - Little available QoS parameters
- Switched Fabrics (RACEway, RapidIO, Myrinet, Infiniband)
  - *No* hardware interrupts
  - No media contention (N x N traffic)
  - QoS is priority for resolving contention if N is exceeded

ORB*express* RT: A QoS-based, Real-time ORB

- Allows full control of QoS on user defined transports on a per-connection basis
- Blends user control of
  - Processor scheduling and
  - Communications channel scheduling
- Allows for any processor and communications resource scheduling algorithm
- Supports wide range of optimization criteria
  - Bounded latency (hard real-time) systems
  - High throughput, cost constrained soft real-time systems
- Introduced Real-Time QoS Interceptors™
  - Give full control to the developer to manage the ORBs use of QoS and each communication channel
  - Default interceptor conforms to Real-time CORBA 1.0 specification

User Replaceable Transports with QoS

- ORB*express* GT was the first ORB with user replaceable transports with or in lieu of TCP/IP
- ORB*express* RT evolved this architecture for replaceable transports by allowing application developers to define new classes of QoS as supported by their plug-in transport

Separating Application Design from Scheduling

- ORBs are a golden opportunity for managing application scheduling
- Requires access to and control of
  - Threads
  - Resource management
  - Communication channels
- As a result ORB*express* RT allows developers to separate
  - application design from
  - application scheduling

Portability Across Communication Protocols

- Combining
  - Management of processor availability and
  - Communications QoS
- Yields improved ability to port an application among differing communications protocols
- Porting then involves
  - Easy porting of source code
  - Completely new scheduling analysis

Summary

- Scheduling communications resources is important
- Coordinating the scheduling of communications resources with processor resources is essential
- Products exist that allow for this coordinated scheduling

Further Information

- Real-time CORBA 1.0
  - OMG CORBA 2.6.1 specification, chapter 24
- Real-time CORBA 2.0: Dynamic Scheduling
  - http://www.omg.org/cgi-bin/doc?ptc/01-08-34.pdf
- Tests of various communication technologies
  - Lockheed Martin Advanced Technology Labs' Agent and Distributed Objects Quality of Service (QoS)
  - http://www.atl.external.lmco.com/projects/QoS/
- Information about CORBA for Real-Time, Embedded, and High Performance Applications
  - http://www.ois.com/resources/corb-1.asp
- Real-time and embedded CORBA discussion forum
  - http://www.realtime-corba.com