

Timeliness meets QoS

What should we do?

Dock Allen

The MITRE Corporation

QoS for Timeliness

- Res Temporalis
 - time sensitive (window of validity)
 - **time critical** (deadlines, possibly closed loop controls)
 - **temporal pacing** (data delivered at its real-world pace with a known delay)
 - temporal ordering (from a single source)
 - temporal coherence (from multiple sources)
 - response to temporal relationships (workflow)
- people USUALLY mean “time critical” or “temporal pacing” when they say Real-time

Timeliness and QoS

- Timeliness is an End-to-End Characteristic
- QoS is a useful commercial concept that we can fit “timeliness” into (to take advantage of commercial frameworks)
- “End-to-end” timeliness requires:
 - Supporting temporal QoS at all levels
 - Managing QoS across infrastructure boundaries
 - middleware
 - OS
 - networks
 - A dynamic view of resource management

State of the Practice

- QoS-based timeliness support (language, middleware)
 - CORBA (Real-time CORBA 1.0 and Real-time CORBA 2.0: Dynamic Scheduling)
 - Real-time Specification for Java (Sun)
 - J-Consortium Real-time Java Specification
- UML extensions for Real-time and QoS
- Operating System Support (limited to priority-based scheduling in most cases)
 - MK7 and Alpha research OSs supported more flexible scheduling
- Network support is mostly limited to QoS for I/O
 - Fibre Channel has a limited priority concept

Interesting Research

- MITRE research on mapping application-QoS onto QoS-based scheduling
 - would be useful to repeat with a commercial application
- Research OSs that support QoS-based scheduling
 - Alpha and MK7
- Darpa (and other) work on Adaptive Resource Management

What's Missing

- Useful ways of expressing application-level QoS
- Better understanding of how to map “application-level QoS” onto the available mechanisms in the infrastructure
- Approaches for mapping QoS between different infrastructure domains / levels (e.g. from CORBA to a network)
- OS-level support for dynamic scheduling
- Support for *temporal* QoS (TQoS) at the network level

Possible Tasks

- Analyze a commercial application (what is its application QoS? how does this map to scheduling QoS and network QoS?)
- Prototype QoS-scheduling in Linux
- Define standard terms for OS-level TQoS and network level TQoS
 - Define metrics for runtime monitoring of TQoS
- Investigate approaches for top-to-bottom mapping / translation functions

- Reverse engineer FedEx
- Boeing supplier management process
- Mark, Sally, Dock, etc

- Classify applications by application patterns?
- Issue: policy definition approach???
- Need near term, mid term, long term goals
- Russ, Mark, Dock, Jean, Doug, John, Wayne

Suggested Roadmap

Analyze QoS for a commercial application (worked example # 2)

Standard terms for OS and network TQoS

Standard metrics for OS and network TQoS

Linux prototype of QoS Scheduler?

Mapping / translation functions

