Architectural Considerations for Real-Time CORBA ORBs and Applications

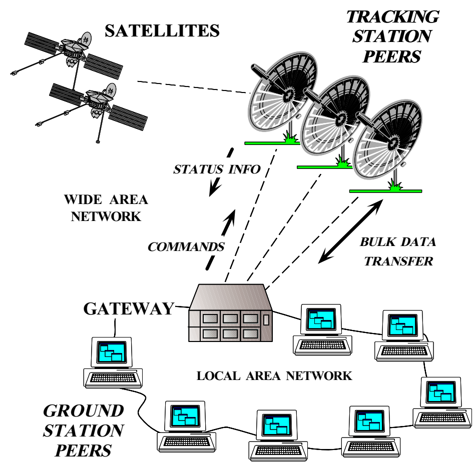
David L. Levine
Washington University, St. Louis
levine@cs.wustl.edu

19 May 1998

http://www.cs.wustl.edu/∼levine/research/spartan98.pdf

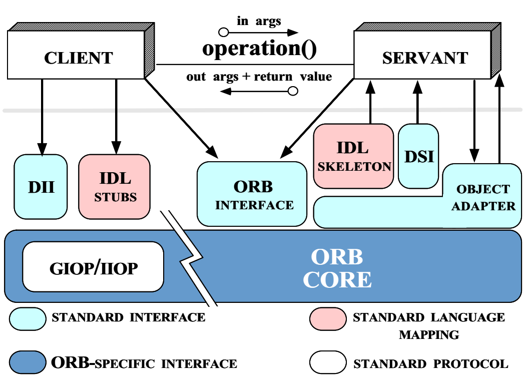
Sponsors: Sprint, Siemens MED and ZT, OTI, NSF grant NCR-9628218, GDIS, DARPA contract 9701516, and Boeing

Motivation for Real-time Middleware



- Many applications require QoS guarantees
  - *e.g.*, telecom, avionics, WWW
- Existing middleware doesn't support QoS effectively
  - *e.g.*, CORBA, DCOM, DCE
- Solutions must be *integrated*
  - *Vertically* and *horizontally*

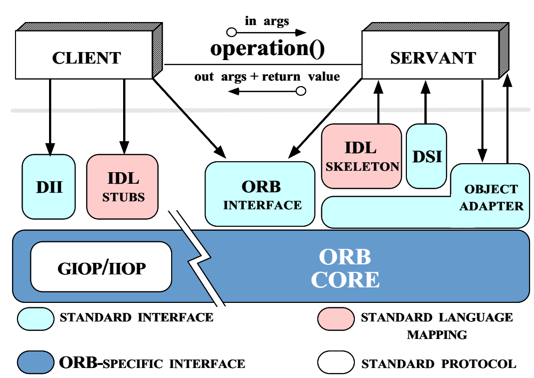
Candidate Solution: CORBA



www.cs.wustl.edu/∼schmidt/corba.html

- **Goals of CORBA**
  - Simplify distribution by automating
    - ∗ Object location and activation
    - ∗ Parameter marshaling
    - ∗ Demultiplexing
    - ∗ Error handling
  - Provide foundation for higher-level services

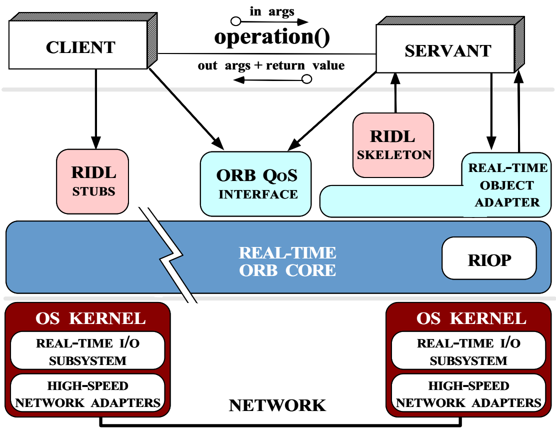
Limitations of CORBA for Real-time Systems



www.cs.wustl.edu/∼schmidt/
ORB-endsystem.ps.gz

- **Limitations**
  - Lack of QoS specifications
  - Lack of QoS enforcement
  - Lack of real-time programming features
  - Lack of performance optimizations

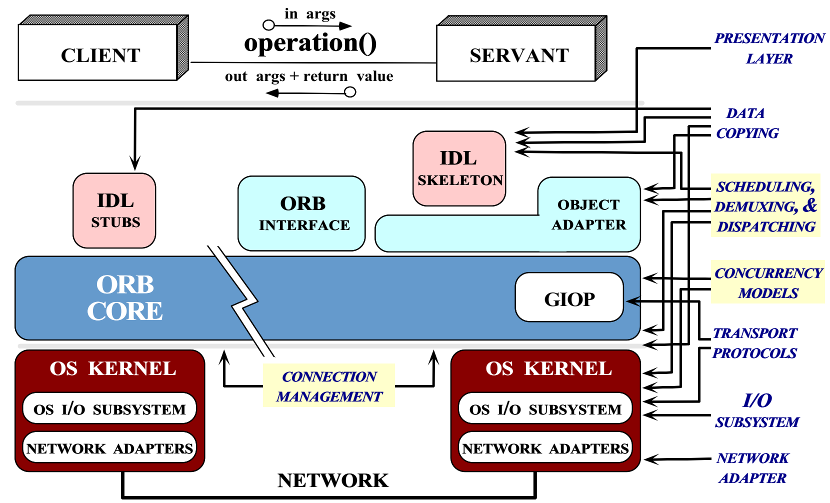
The ACE ORB (TAO)



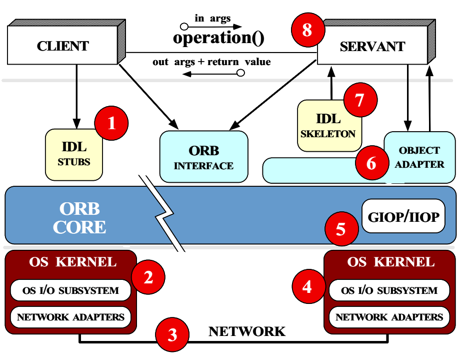
www.cs.wustl.edu/∼schmidt/TAO.html

- **TAO Overview**
  - A high-performance, real-time ORB
    - ∗ Telecom and avionics focus
  - Leverages the ACE framework
    - ∗ Runs on RTOSs, POSIX, and Win32
- **Related work**
  - U. RI/MITRE
  - ARMADA, U. Mich.
  - QuO, BBN

Scope: Real-time Features and Optimizations in TAO



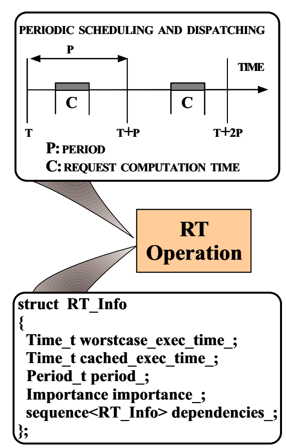
Problem: Meeting End-to-End QoS Requirements



1) CLIENT MARSHALING
2) CLIENT PROTOCOL QUEUEING
3) NETWORK DELAY
4) SERVER PROTOCOL QUEUEING
5) THREAD DISPATCHING
6) REQUEST DISPATCHING
7) SERVER DEMARSHALING
8) METHOD EXECUTION

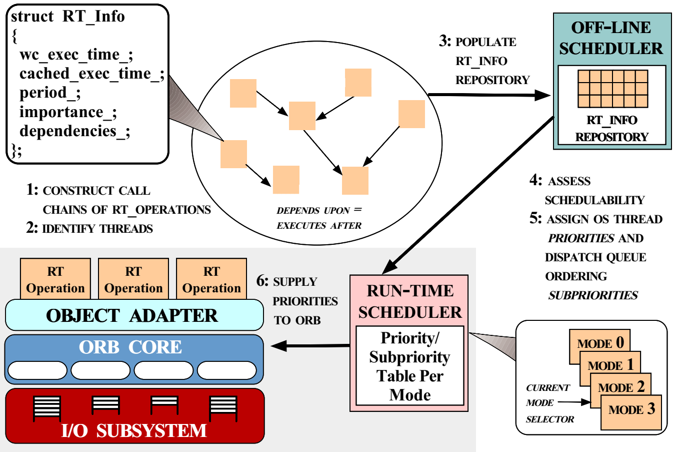
- **Design Challenges**
  - Specifying QoS requirements
  - Meeting operation scheduling deadlines
  - Alleviating priority inversion and non-determinism
  - Reducing latency/jitter for demultiplexing

Problem: Providing QoS to CORBA Operations



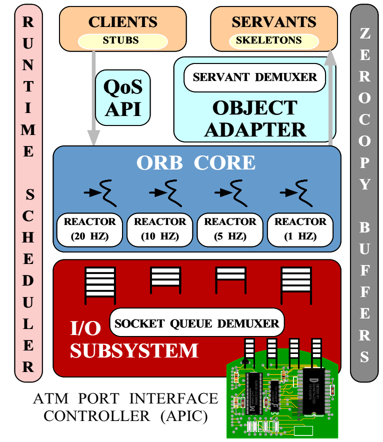
- **Design Challenges**
  - Specifying/enforcing QoS requirements
  - Focus on *Operations* upon *Objects*
    - ∗ Rather than communication channels or threads/synchronization
- **Initial focus**
  - Static scheduling
  - Non-distributed
- **Solution Approach**
  - Servants publish resource, *e.g.*, CPU, requirements and (periodic) deadlines
  - Most clients are also servants

Solution: TAO's Real-time Static Scheduling Service



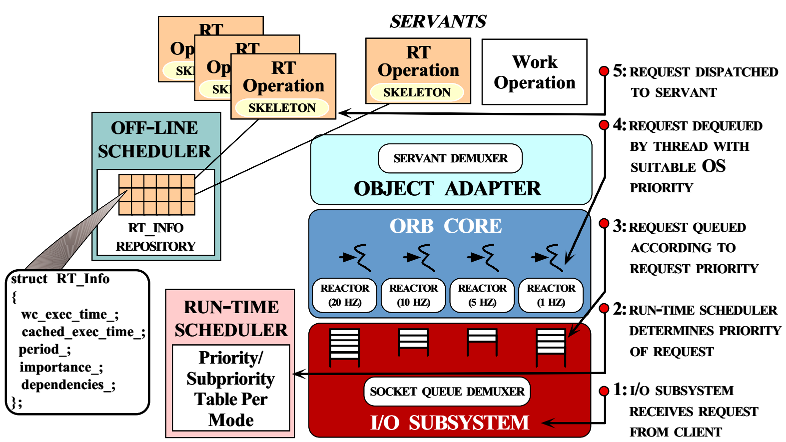
www.cs.wustl.edu/∼schmidt/TAO.ps.gz

Solution: TAO's Real-time ORB Endsystem

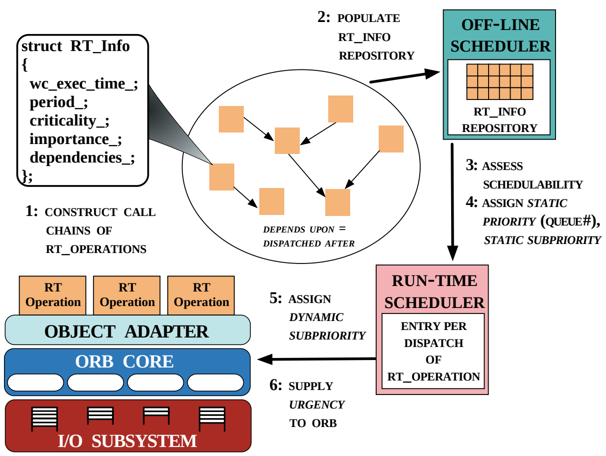


- **Solution Approach**
  - Integrate RT dispatcher into ORB endsystem
  - Support multiple request scheduling strategies
    - ∗ *e.g.*, RMS, EDF, and MUF
  - Requests ordered across thread priorities by OS dispatcher
  - Requests ordered *within* priorities based on *data dependencies* and *importance*

Real-time ORB Endsystem Use-case

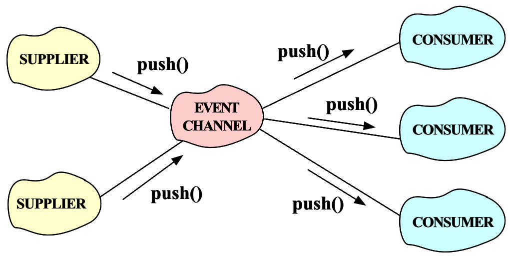


TAO's Real-time Dynamic Scheduling Service



www.cs.wustl.edu/∼schmidt/dynamic.ps.gz

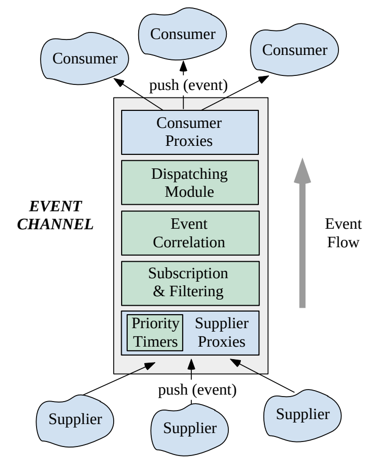
COS Event Service



www.cs.wustl.edu/~schmidt/report-doc.html

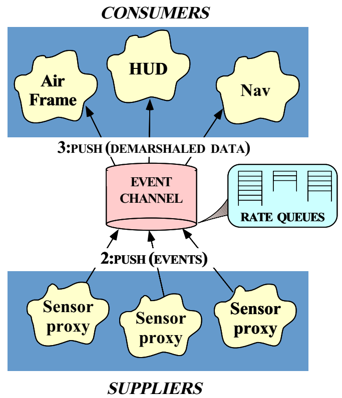
- **Features**
  - Decoupled consumers and suppliers
  - Transparent group communication
  - Asynchronous communication
  - Abstraction for distribution
  - Abstraction for concurrency

TAO's Event Service

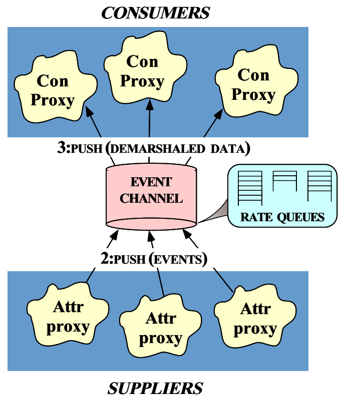


- **Features**
  - Stream-based architecture
    - Enhanced pluggability
  - Subscription/filtering
    - Source and type-based filtering
  - Event correlations
    - Conjunctions (A+B+C)
    - Disjunctions (A|B|C)
  - Real-time scheduling support
    - Priority-based dispatching
    - Priority-based preemption
    - Interval timeouts
    - Deadline timeouts

RT Event Channel Use-cases

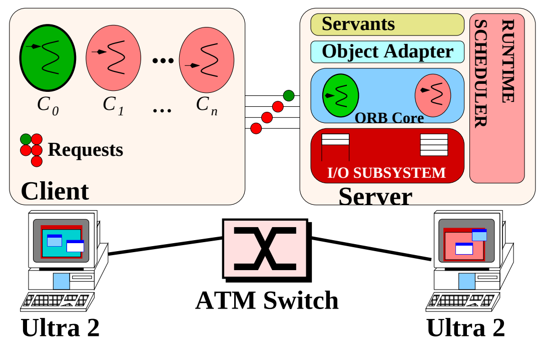


Avionics



Network management

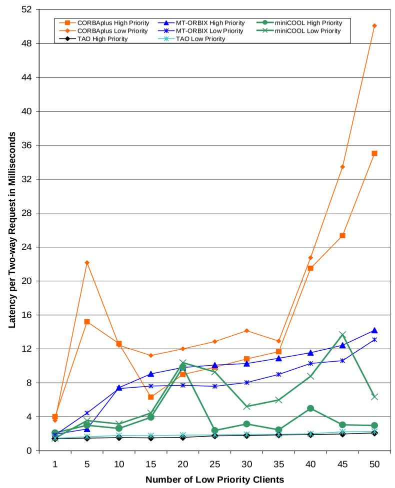
Priority Inversion Experiments



www.cs.wustl.edu/~schmidt/RT-perf.ps.gz

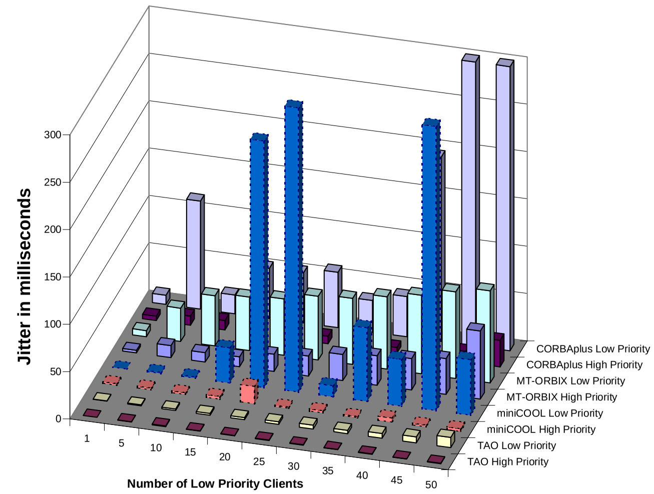
- One high-priority client
- *1..n* low-priority clients
- Server factory implements *thread-per-priority*
  - *Highest* real-time priority for high-priority client
  - *Lowest* real-time priority for low-priority clients

ORB Latency and Priority Inversion Results



- **Synopsis of results**
  - COOL's latency is lower for small # of clients
  - TAO's latency is lowest for large # of clients
  - TAO avoids priority inversion
    - * *i.e.*, high priority client always has lowest latency

ORB Jitter Results



- **Definition**
  - Variance from average latency
- **Synopsis of results**
  - TAO's jitter is lowest and most consistent
  - MT-Orbix's jitter is highest and more variable

Concluding Remarks

- Developers of distributed applications confront recurring challenges that are largely application-independent
  - *e.g.*, service initialization and distribution, error handling, flow control, scheduling, event demultiplexing, concurrency control, persistence, fault tolerance
- Successful developers resolve these challenges by applying appropriate *design patterns* to create communication *frameworks* and components
- ORBs are an effective way to achieve reuse of distributed software components
- The next generation of ORBs will provide much better support for real-time QoS

For Further Information

- These slides: `http://www.cs.wustl.edu/~levine/research/spartan98.pdf`
- More detail on TAO: `http://www.cs.wustl.edu/~schmidt/RT-ORB.ps.gz`
- TAO Event Channel: `http://www.cs.wustl.edu/~levine/research/JSAC98.ps.gz`
- TAO static scheduling: `http://www.cs.wustl.edu/~schmidt/TAO.ps.gz`
- TAO dynamic scheduling: `http://www.cs.wustl.edu/~levine/research/scheduling/dynamic.pdf`
- ORB Endsystem Architecture: `http://www.cs.wustl.edu/~schmidt/RT-middleware.ps.gz`

For Further Information

- Performance Measurements:
  - Demultiplexing latency: http://www.cs.wustl.edu/~schmidt/GLOBECOM-97.ps.gz
  - SII throughput: http://www.cs.wustl.edu/~schmidt/SIGCOMM-96.ps.gz
  - DII throughput: http://www.cs.wustl.edu/~schmidt/GLOBECOM-96.ps.gz
  - Latency, scalability: http://www.cs.wustl.edu/~schmidt/ICDCS-97.ps.gz
  - IIOP: http://www.cs.wustl.edu/~schmidt/IIOP.ps.gz
- More detail on CORBA: http://www.cs.wustl.edu/~schmidt/corba.html
- ADAPTIVE Communication Environment (ACE):
  http://www.cs.wustl.edu/~schmidt/ACE.html