

High-performance, Real-time CORBA ORBs for ATM Networks

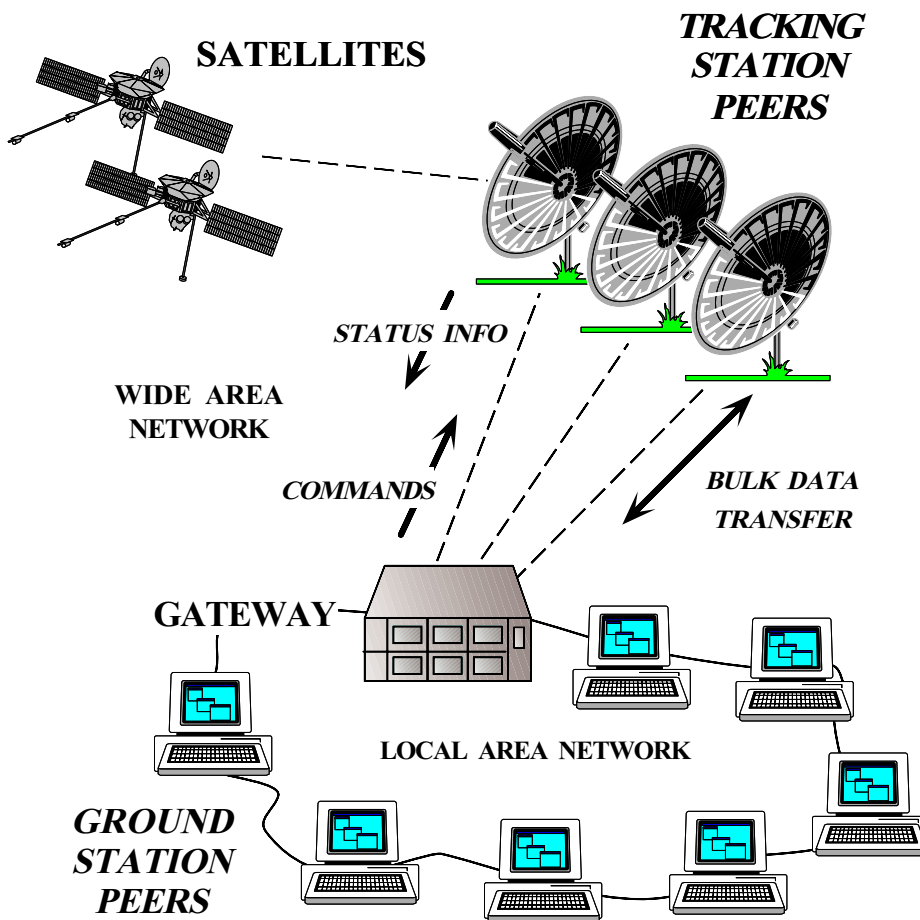
Douglas C. Schmidt
schmidt@cs.wustl.edu

Washington University, St. Louis
www.cs.wustl.edu/~schmidt/TAO.html

Sponsors

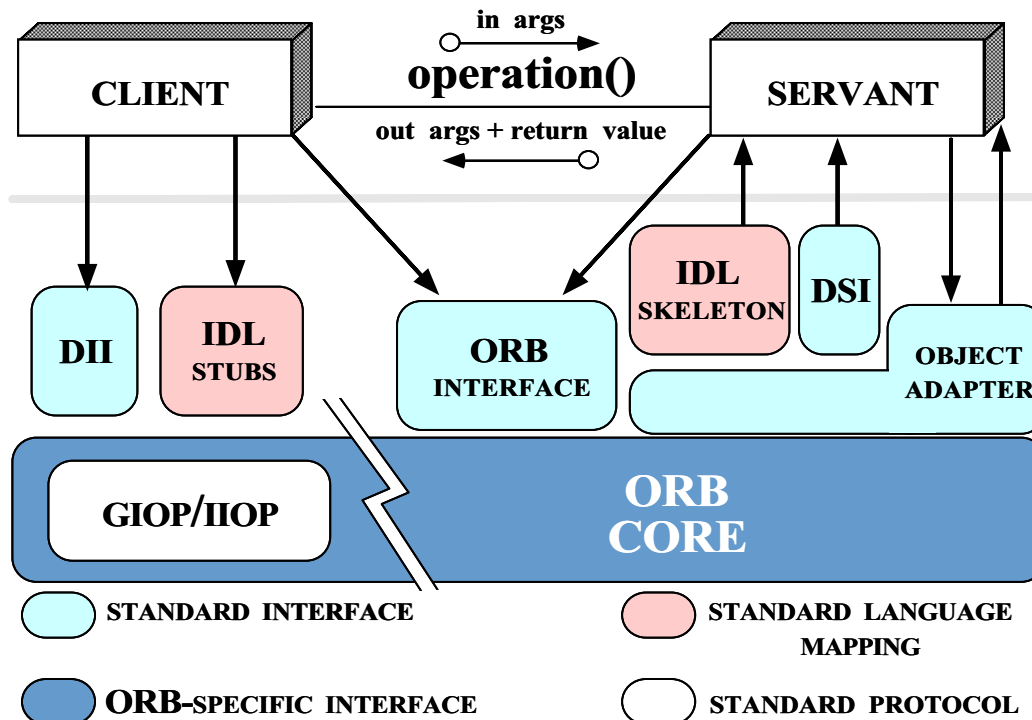
NSF, DARPA, Bellcore, Boeing, CDI,
Kodak, Lucent, Motorola, OTI, SAIC,
Siemens SCR, Siemens MED, Siemens ZT, Sprint

Problem: Lack of Real-time Middleware



- Many applications require QoS guarantees
 - e.g., telecom, avionics, WWW
- Building these applications manually is hard
- Existing middleware doesn't support QoS effectively
 - e.g., CORBA, DCOM, DCE
- Solutions must be *integrated*

Candidate Solution: CORBA



www.cs.wustl.edu/~schmidt/corba.html

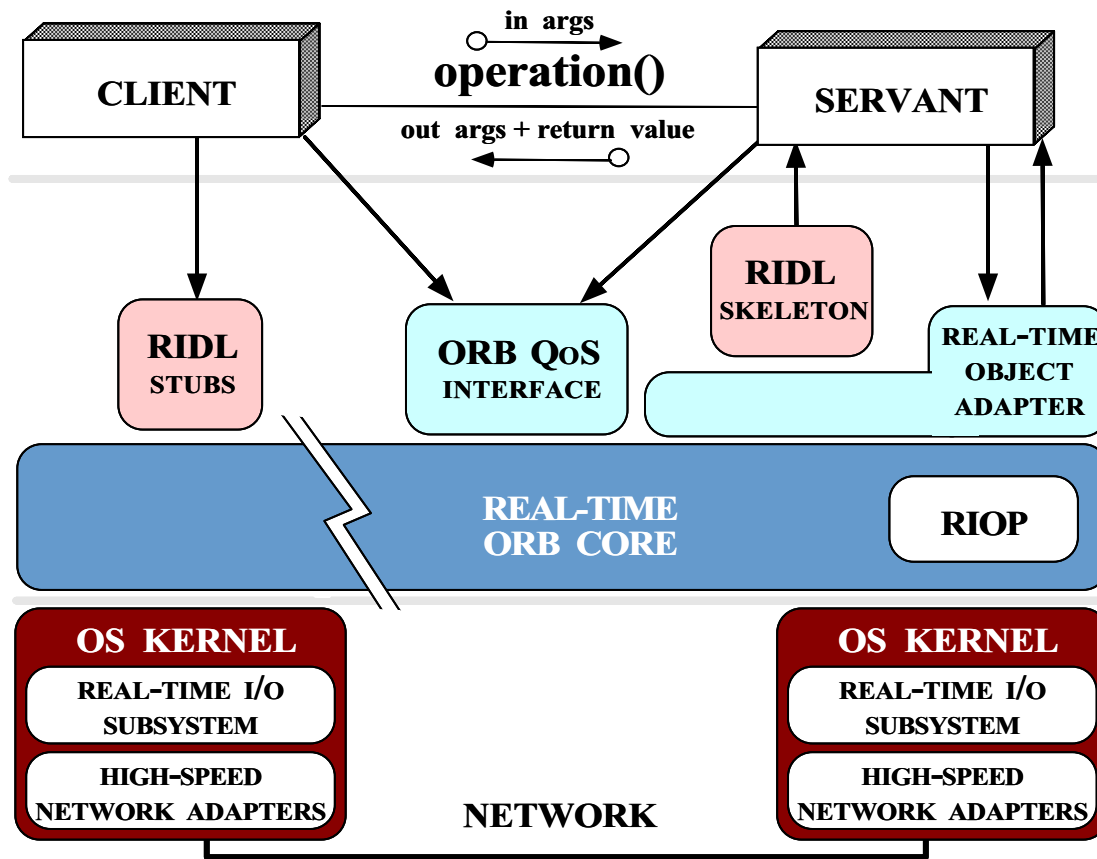
• Goals of CORBA

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

Motivation for CORBA

- Simplifies application interworking
 - CORBA provides higher level integration than traditional *untyped TCP bytestreams*
- Provides a foundation for higher-level distributed object collaboration
 - e.g., Windows OLE and the OMG Common Object Service Specification (COSS)
- Benefits for distributed programming similar to OO languages for non-distributed programming
 - e.g., encapsulation, interface inheritance, and object-based exception handling

The ACE ORB (TAO)



www.cs.wustl.edu/~schmidt/TAO.html

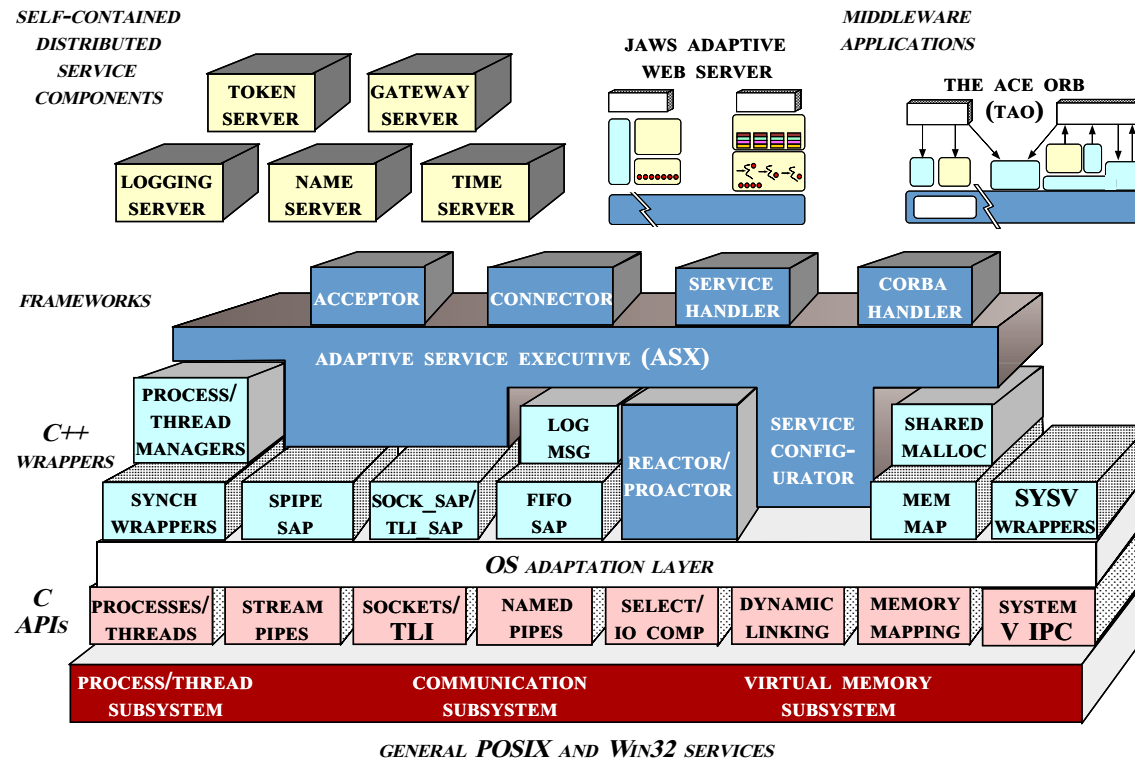
• TAO Overview

- A high-performance, real-time ORB
 - * Telecom and avionics focus
- Leverages the ACE framework
 - * Runs on RTOSs, POSIX, and Win32

• Related work

- QuO at BBN
- ARMADA at U. Mich.

The ADAPTIVE Communication Environment (ACE)



- **ACE Overview**
 - Concurrent OO networking framework
 - Ported to C++ and Java
 - Runs on RTOSs, POSIX, and Win32

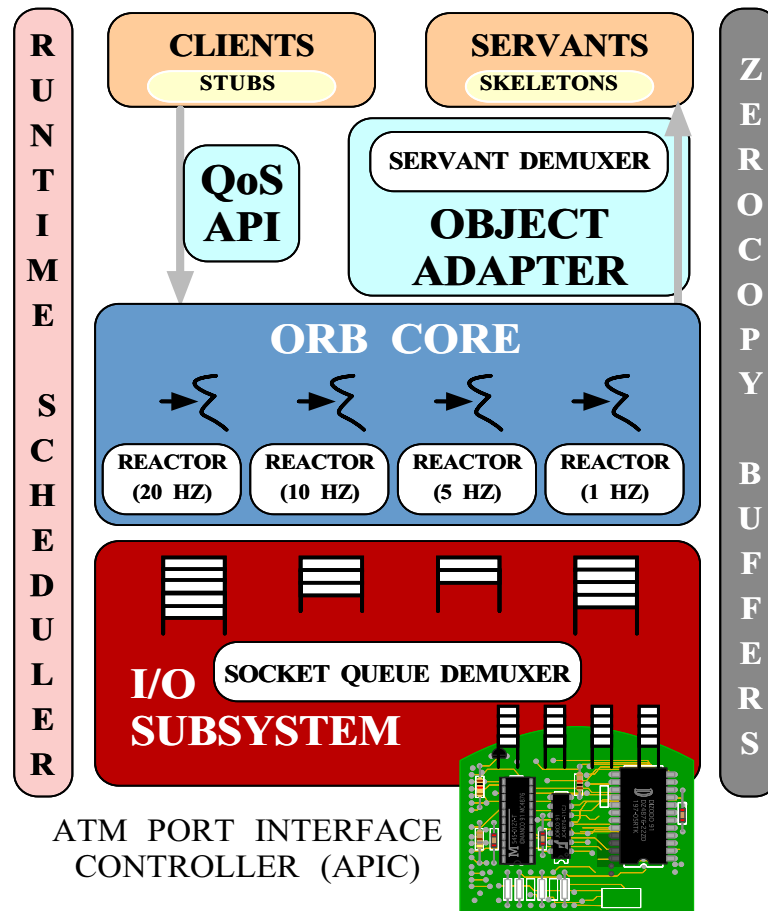
- **Related work**
 - x-Kernel
 - SysV STREAMS

www.cs.wustl.edu/~schmidt/ACE.html

ACE Statistics

- ACE contain $> 125,000$ lines of C++
 - Over 10 person-years of effort
- Ported to UNIX, Win32, MVS, and embedded platforms
 - e.g., VxWorks, LynxOS, pSoS
- Large user community
 - www.cs.wustl.edu/~schmidt/ACE-users.html
- Currently used by dozens of companies
 - Bellcore, Boeing, Ericsson, Kodak, Lucent, Motorola, SAIC, Siemens, StorTek, etc.
- Supported commercially
 - www.riverace.com

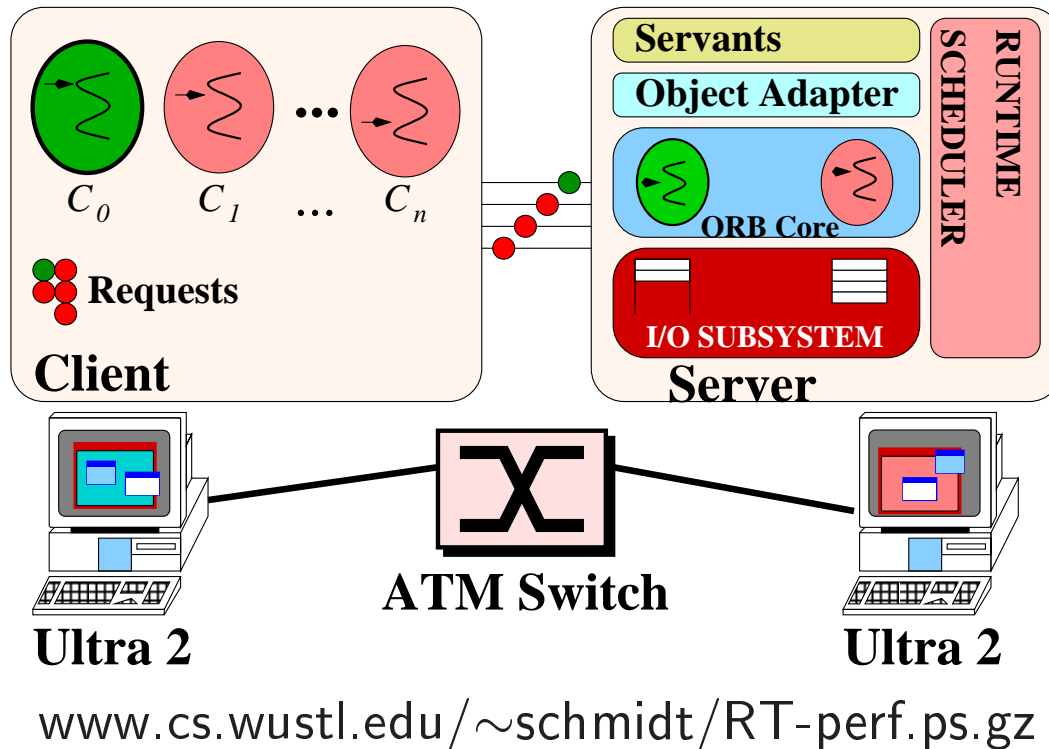
TAO's Real-time ORB Endsysteem Architecture



• Solution Approach

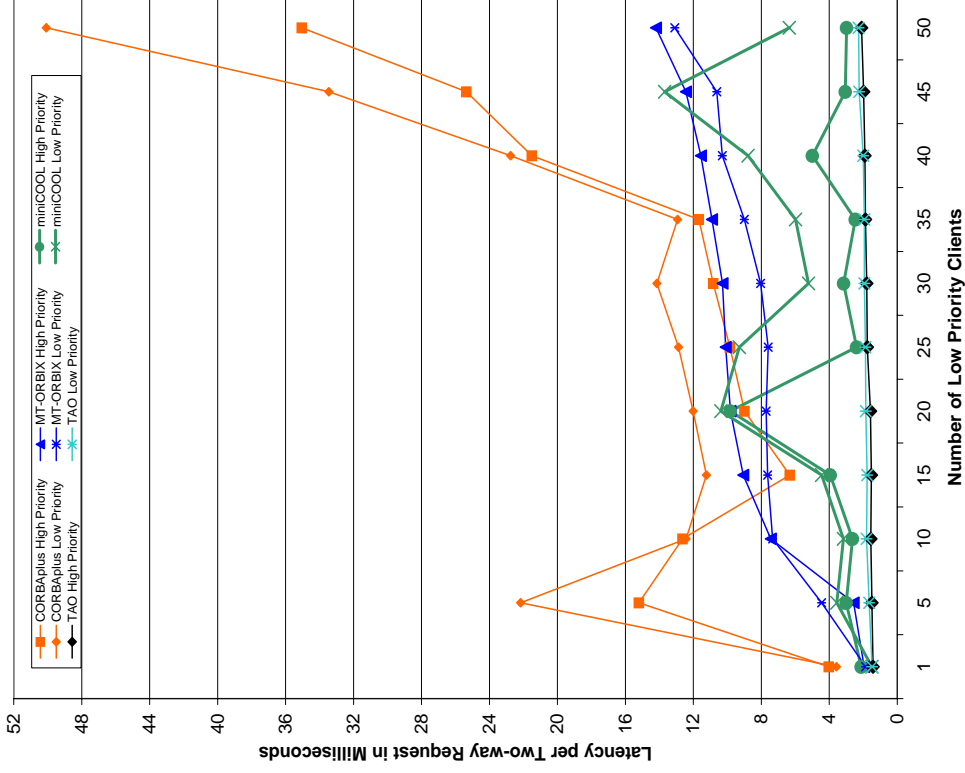
- Integrate RT dispatcher into ORB endsysteem
- Support multiple request scheduling strategies
 - * e.g., RMS, EDF, and MUF
- Requests ordered *across* thread priorities by OS dispatcher
- Requests ordered *within* priorities based on *data dependencies* and *importance*

Real-time Experiments over ATM



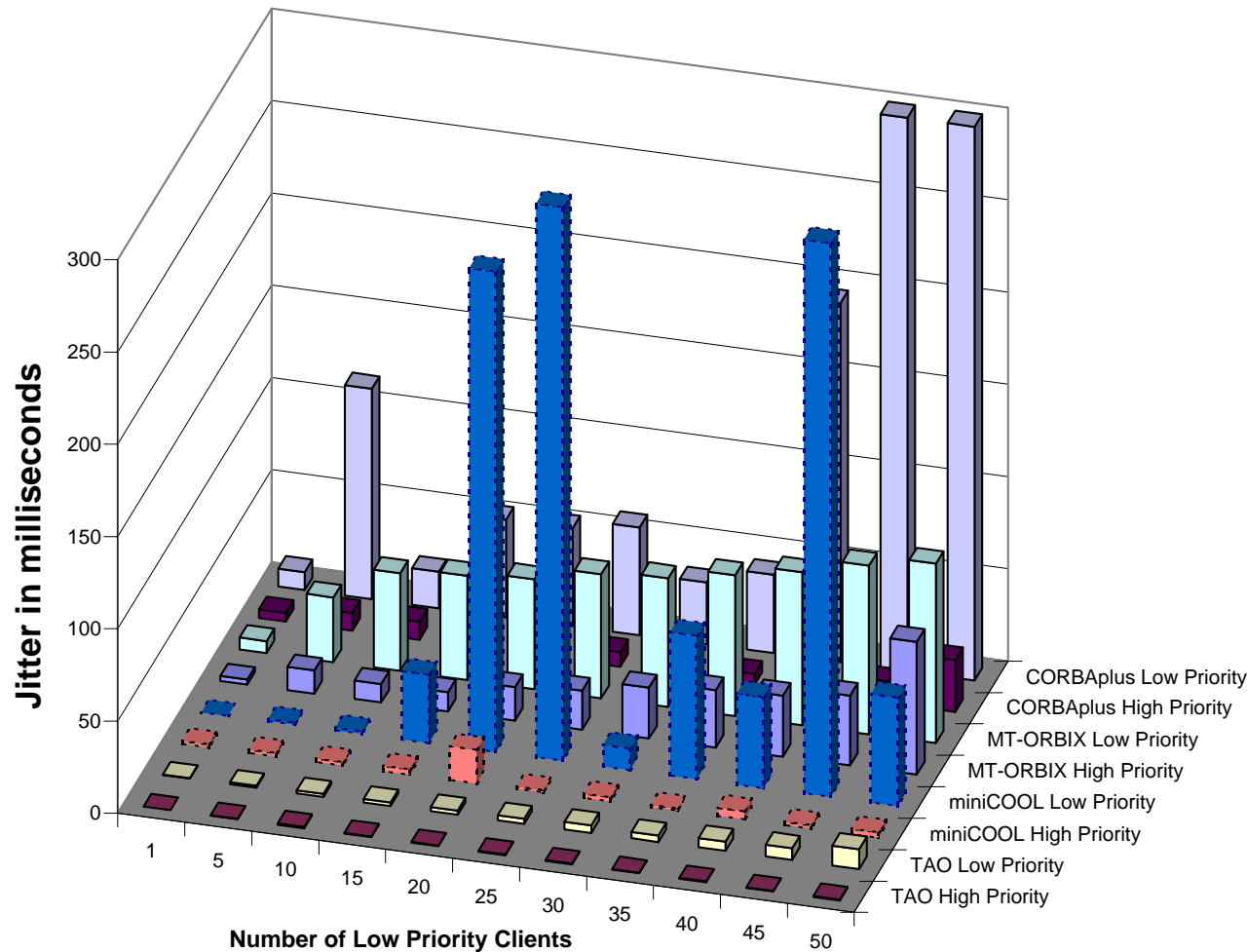
- One high-priority client
- $1..n$ low-priority clients
- Server factory implements *thread-per-priority*
 - *Highest* real-time priority for high-priority client
 - *Lowest* real-time priority for low-priority clients

ORB Latency Results over ATM



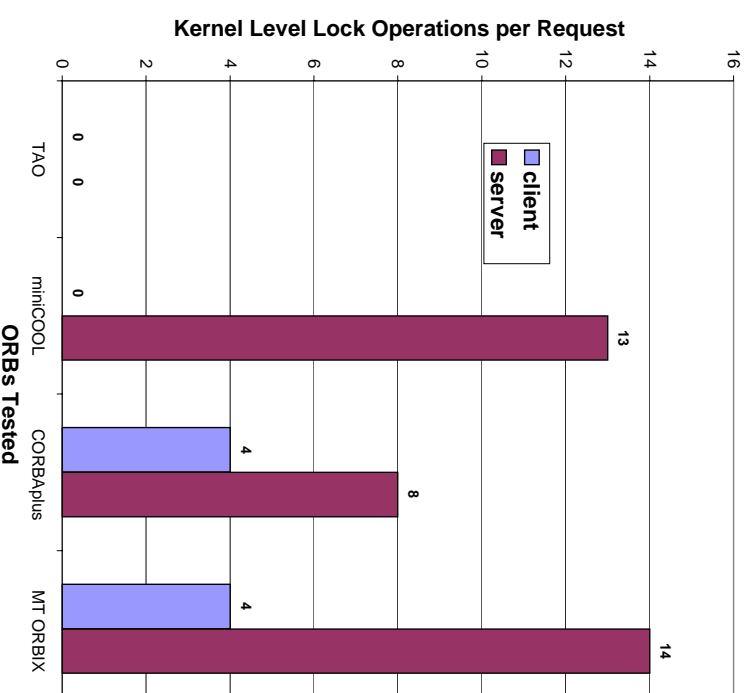
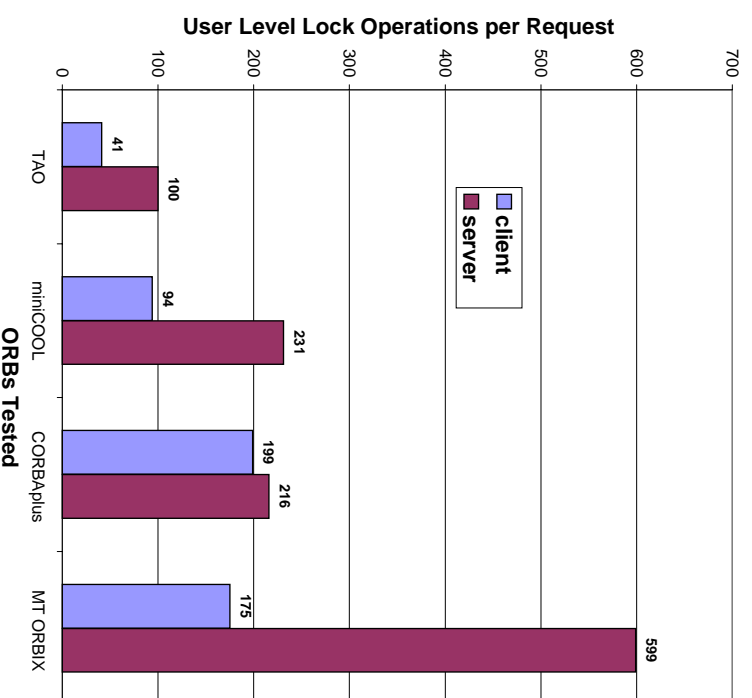
- **Synopsis of results**
 - COOL's latency is lower for small # of clients
 - TAO's latency is lowest for large # of clients
 - TAO avoids priority inversion
 - * *i.e.*, high priority client always has lowest latency

ORB Jitter Results over ATM

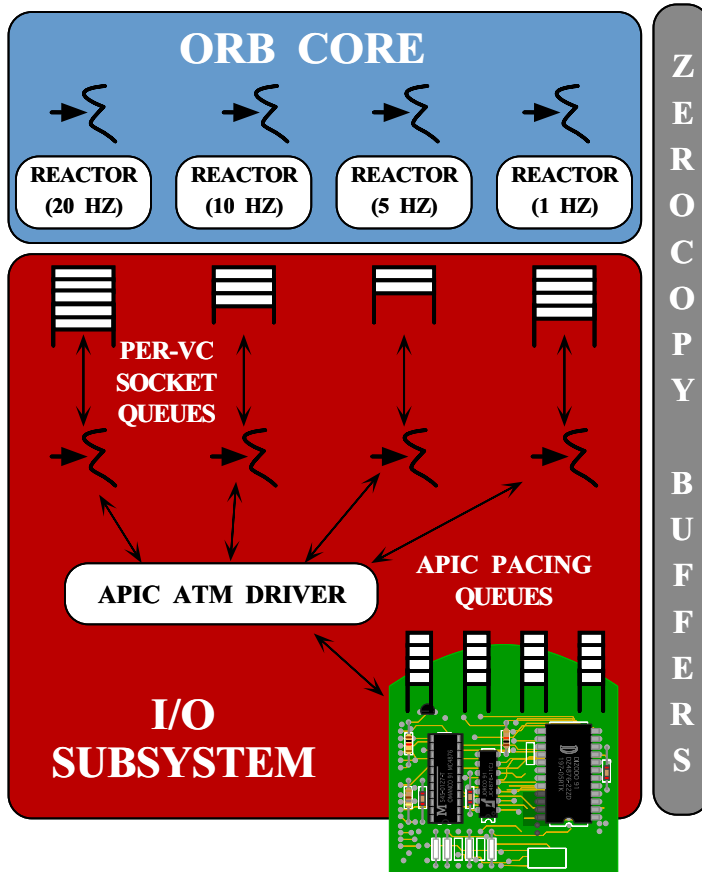


- **Definition**
 - Variance from average latency
- **Synopsis of results**
 - TAO's jitter is lowest and most consistent
 - MT-Orbix's jitter is highest and more variable

User-level and Kernel-level Locking Overhead



Integrating TAO with a Real-time ATM I/O Subsystem



• Key Features

- Vertical integration of QoS through ORB, OS, and ATM network
- Real-time I/O enhancements to Solaris kernel
- Provides rate-based QoS end-to-end
- Leverages APIC features for cell pacing and zero-copy buffering

Concluding Remarks

- Developers of distributed applications confront recurring challenges that are largely application-independent
 - e.g., service initialization and distribution, error handling, flow control, event demultiplexing, concurrency control, persistence, fault tolerance
- Successful developers resolve these challenges by applying appropriate *design patterns* to create communication *frameworks* and *components*
- CORBA 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 over ATM