Applying a Real-time CORBA ORB for Avionics Mission Computing

Douglas C. Schmidt
schmidt@cs.wustl.edu

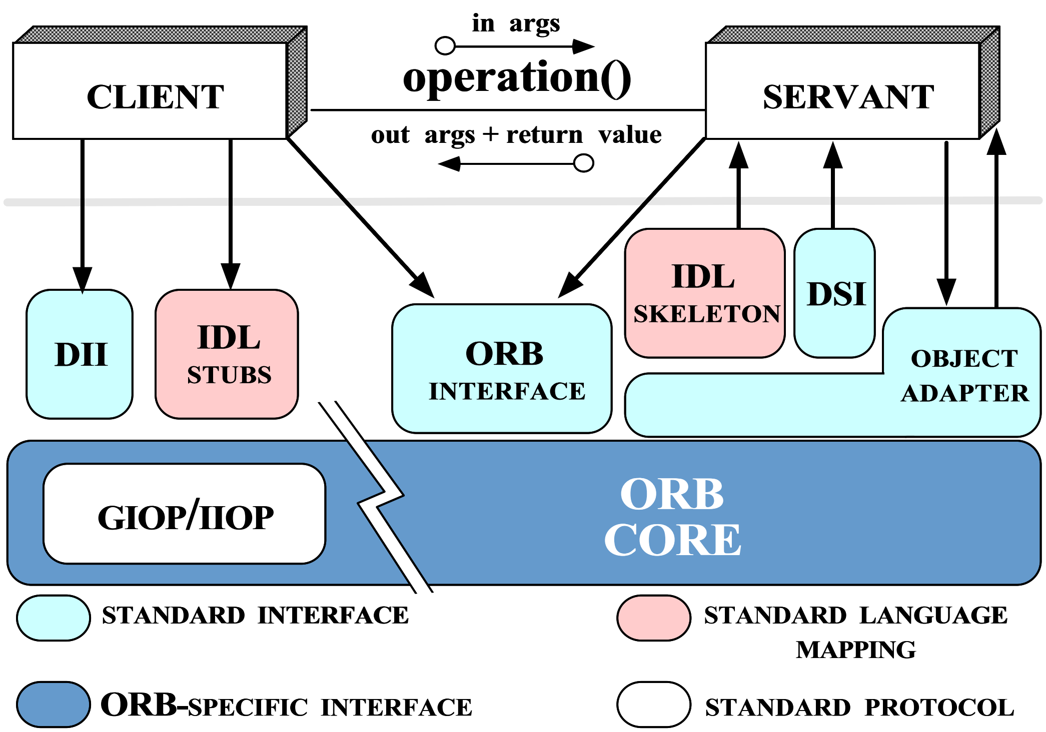
Washington University, St. Louis
www.cs.wustl.edu/∼schmidt/TAO4.ps.gz

Sponsors
Boeing and CDI/GDIS

Mission Computing Design Requirements and Forces

- Integrate real-time scheduling/dispatching in ORB and I/O subsystem for Boeing military aircraft product families
  - *i.e.*, Harrier (AV/8b), F-15, and F/A-18
- Provide all applications with real-time capabilities
  - Both method-oriented and event-oriented applications
- Meet deterministic and statistical QoS requirements
  - *i.e.*, minimize latency, context switching, priority inversion, and non-determinism

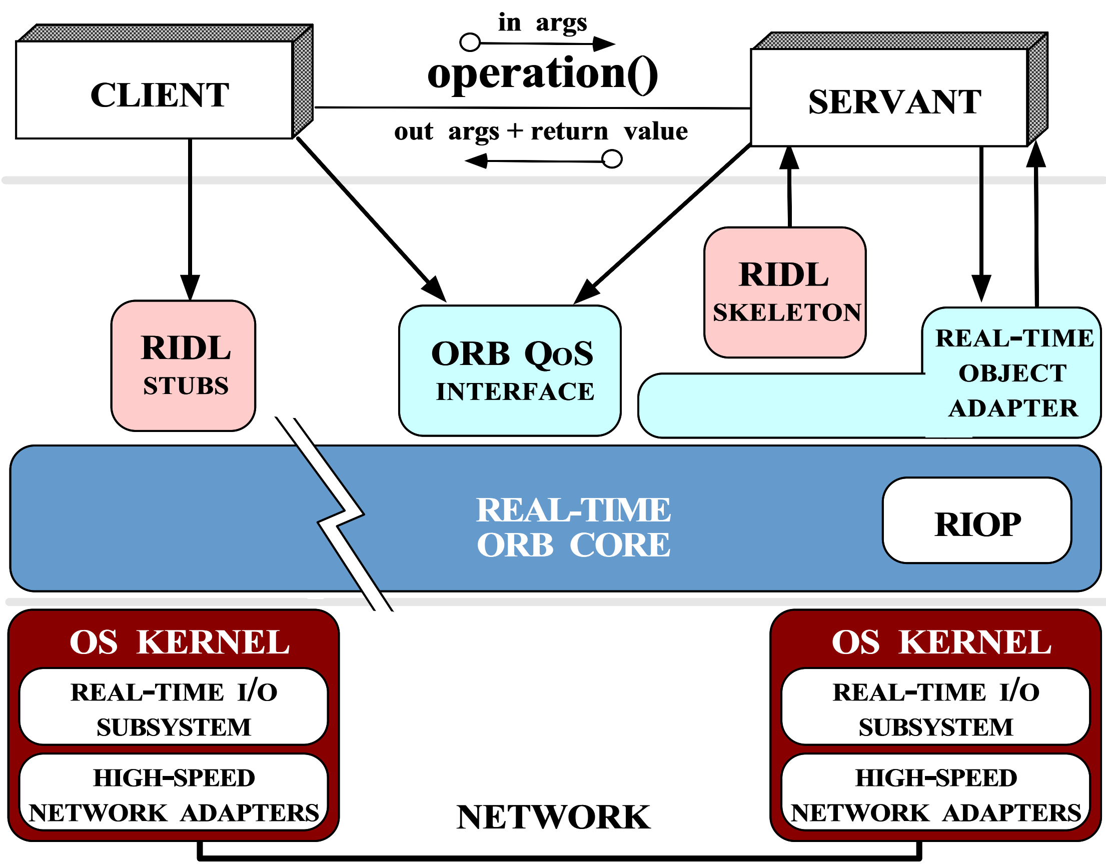
Motivation for CORBA for Mission Computing



www.cs.wustl.edu/∼schmidt/corba.html

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

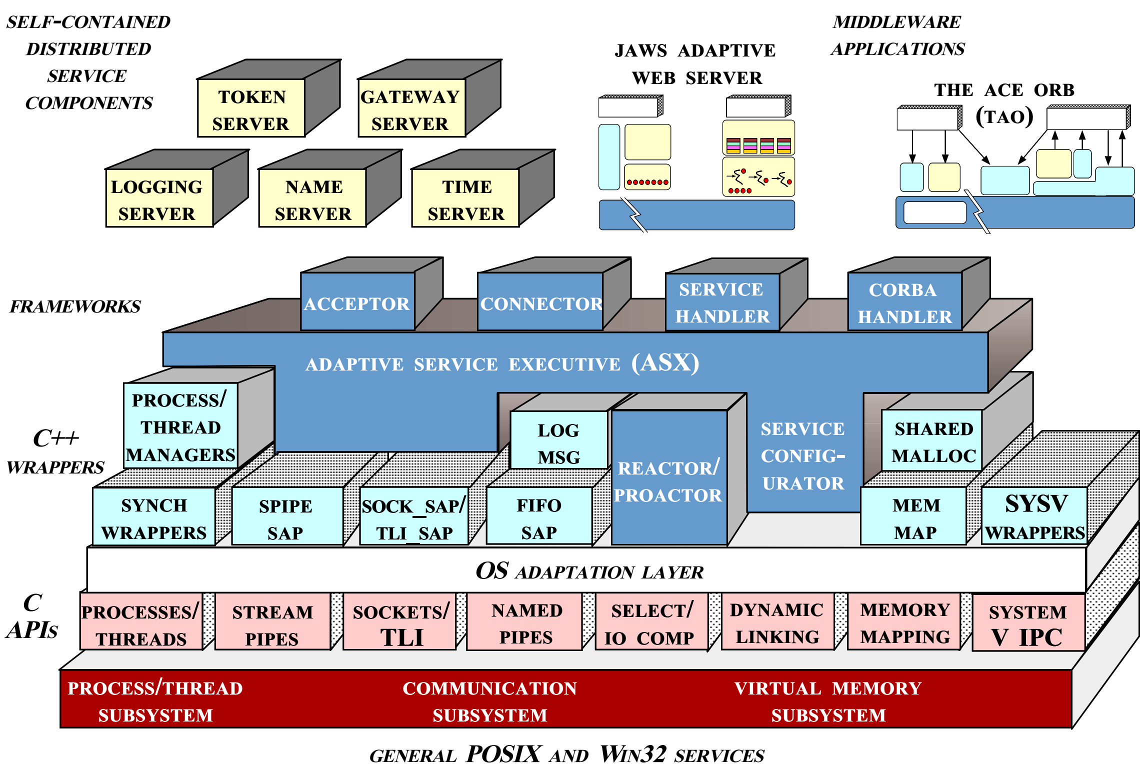
The ACE ORB (TAO)



www.cs.wustl.edu/∼schmidt/TAO.html

- **TAO Overview**
  - A real-time, high-performance ORB
  - Leverages ACE
    - Runs on POSIX, Win32, RTOSs
- **Related work**
  - U. RI, Mitre
  - QuO at BBN
  - ARMADA at U. Mich.

The ADAPTIVE Communication Environment (ACE)



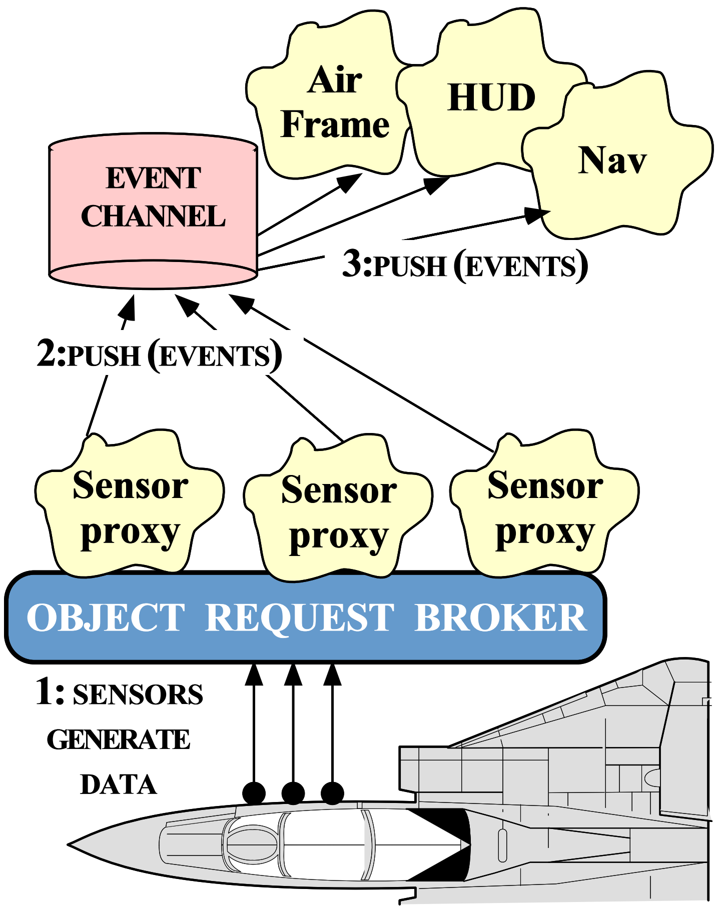
www.cs.wustl.edu/∼schmidt/ACE.html

- **ACE Overview**
  - Concurrent OO networking framework
  - Ported to C++ and Java
  - Runs on RTOSs, POSIX, and Win32
- **Related work**
  - x-Kernel
  - SysV STREAMS

ACE Statistics

- ACE contain $>$ 135,000 lines of C++
  - Over 15 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, Lockheed, Lucent, Motorola, SAIC, Siemens, StorTek, etc.
- Supported commercially
  - www.riverace.com

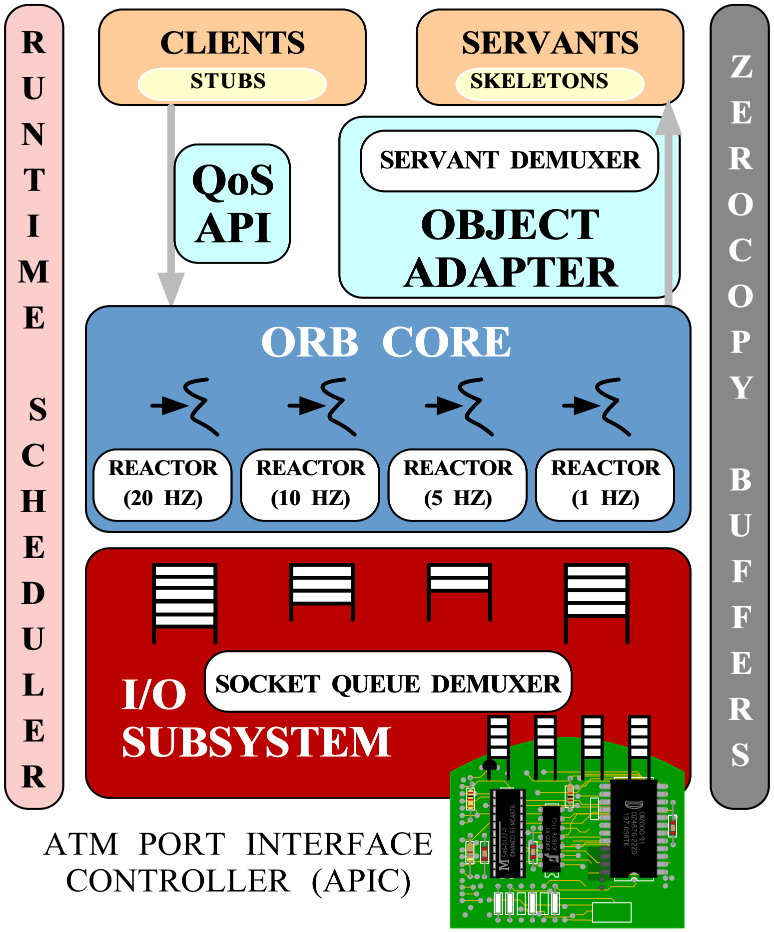
Applying TAO to Avionics Mission Computing



- **Domain Challenges**
  - Periodic deterministic (and some statistical) real-time deadlines
  - COTS infrastructure
  - Open systems
- **Related work**
  - Deng, Liu, and J. Sun ’96
  - Gopalakrishnan and Parulkar ’96
  - Wolfe et al. ’96

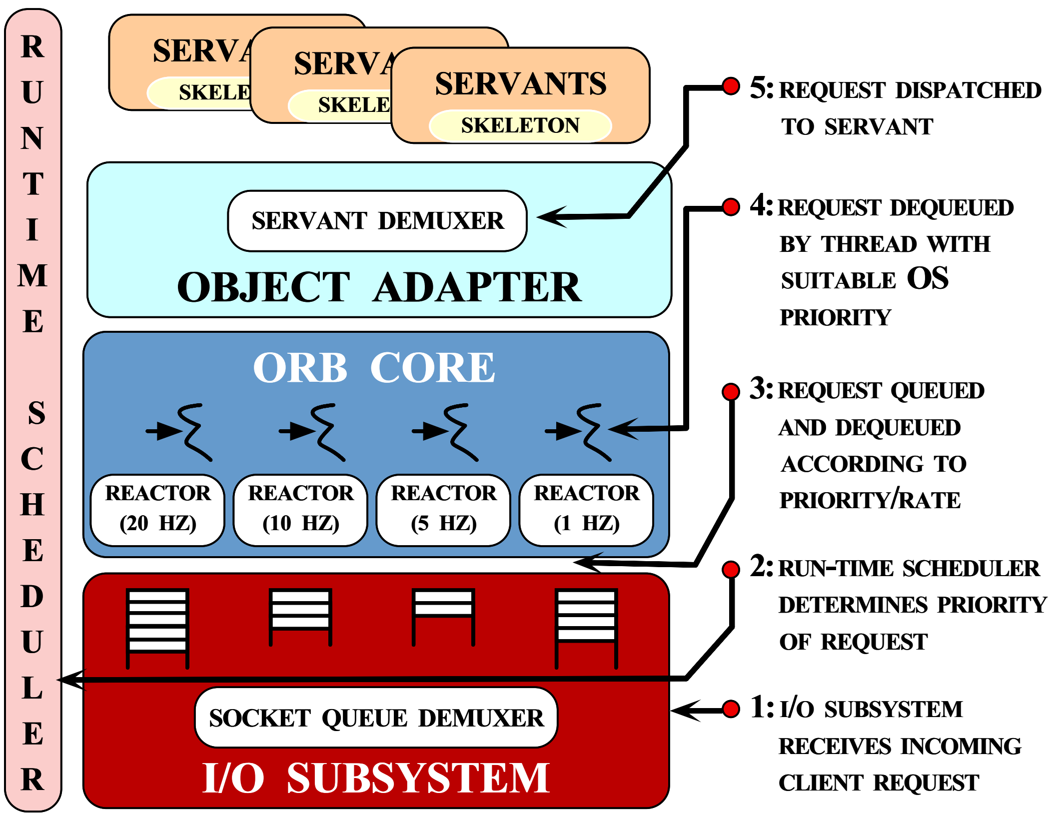
www.cs.wustl.edu/~schmidt/oopsla.ps.gz

TAO's Real-time ORB Endsystem Architecture



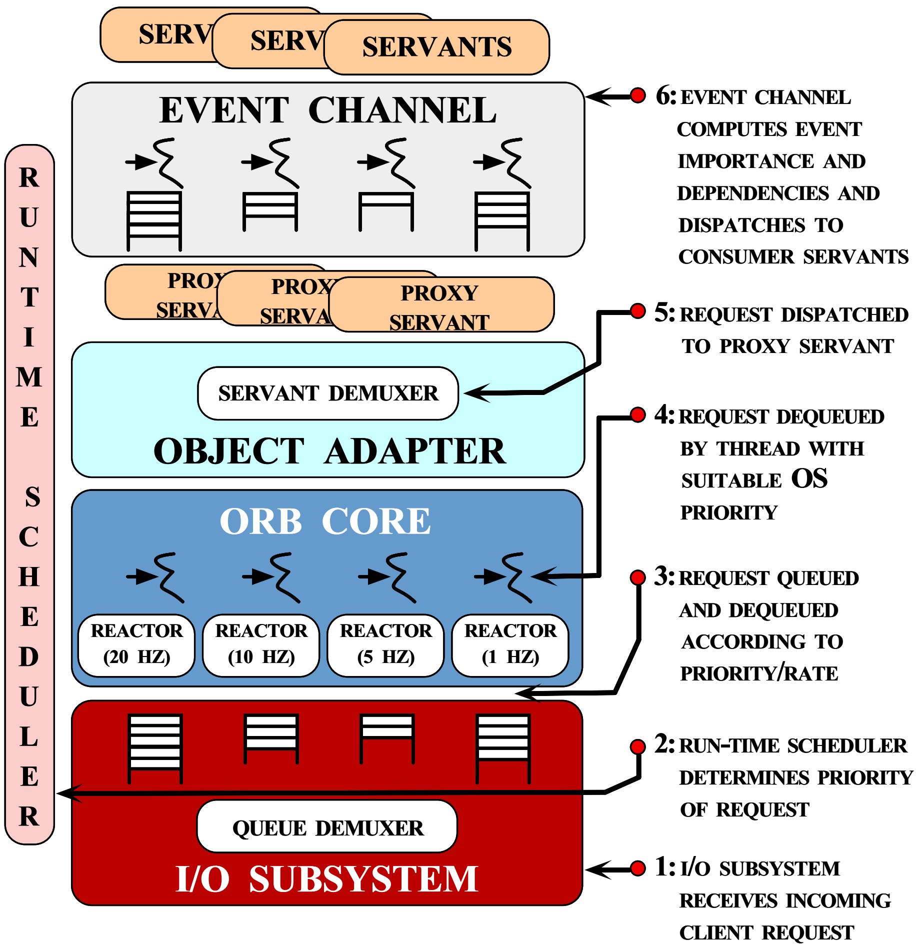
- **Solution Approach**
  - Integrate RT dispatcher into ORB endsystem
  - Support multiple request scheduling strategies
    - *e.g.*, RMS, EDF, and MUF
  - Requests ordered *across* thread priorities by OS dispatcher
  - Requests ordered *within* priorities based on *data dependencies* and *importance*

Server Request Reception Use-case



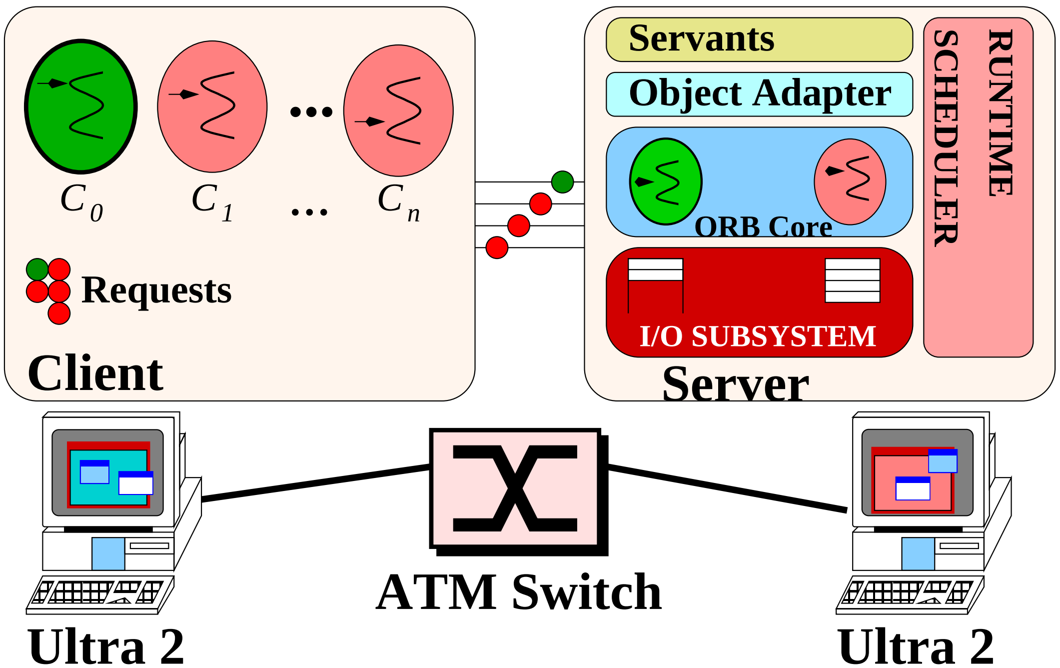
- **Synopsis**
  - I/O subsystem uses port numbers to demux requests to queues and RT threads per rate group
  - A Reactor demuxes/dispatches requests for each rate group

Event Channel Reception Use-case



- **Synopsis**
  - Event Channel threads handle event *importance* and *dependencies*
  - I/O subsystem and ORB Core handle *priorities*

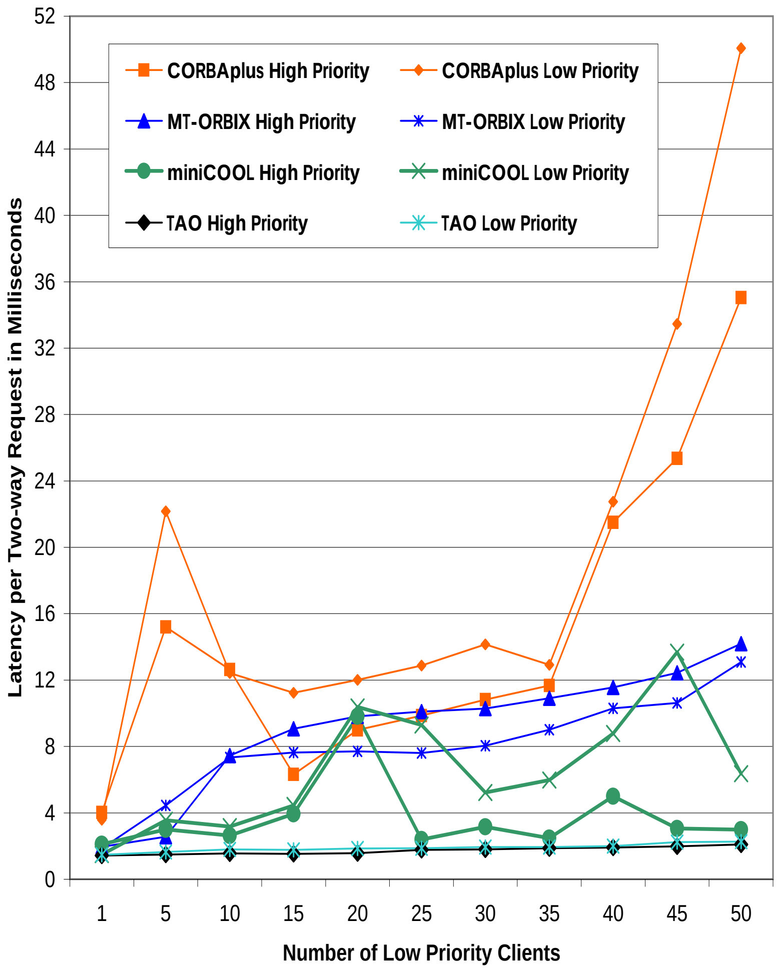
ORB Latency and Priority Inversion Experiments



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

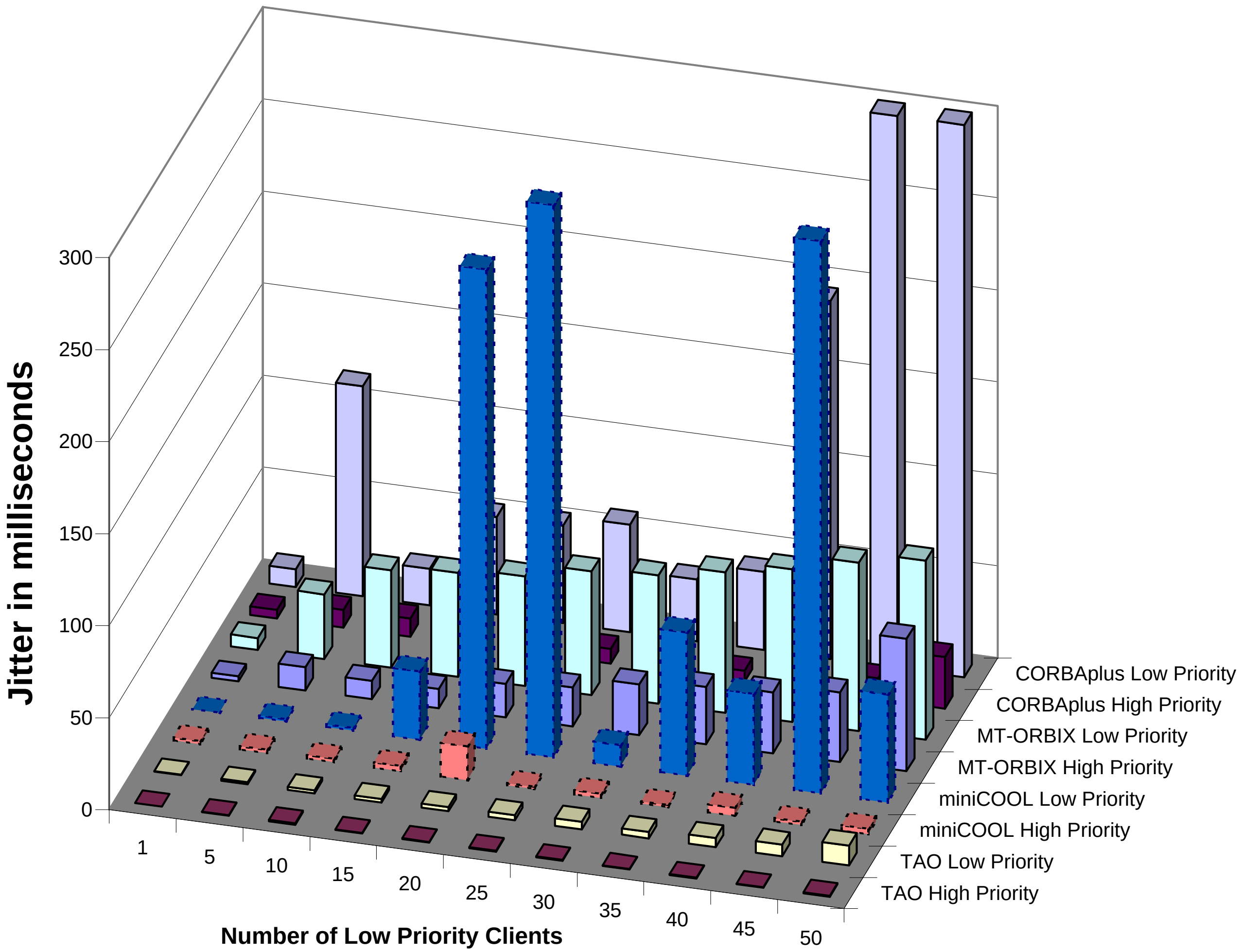
- Vary ORBs, hold OS constant
- Methodology
  - 1 high-priority client
  - *1..n* low-priority clients
  - Server uses *thread-per-priority*
    - *Highest* real-time priority for high-priority client
    - *Lowest* real-time priority for low-priority clients

ORB Latency and Priority Inversion Results



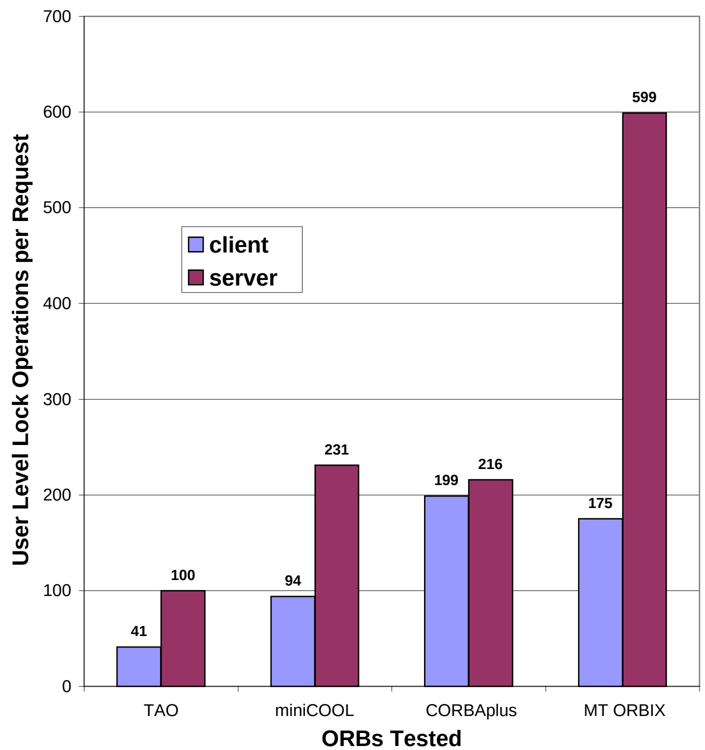
- **Synopsis of results**
  - TAO's latency is lowest
  - TAO avoids priority inversion
    - *i.e.*, high-priority client always has lowest latency
  - Overhead stems from *concurrency* and *connection* architecture
    - *e.g.*, synchronization and context switching

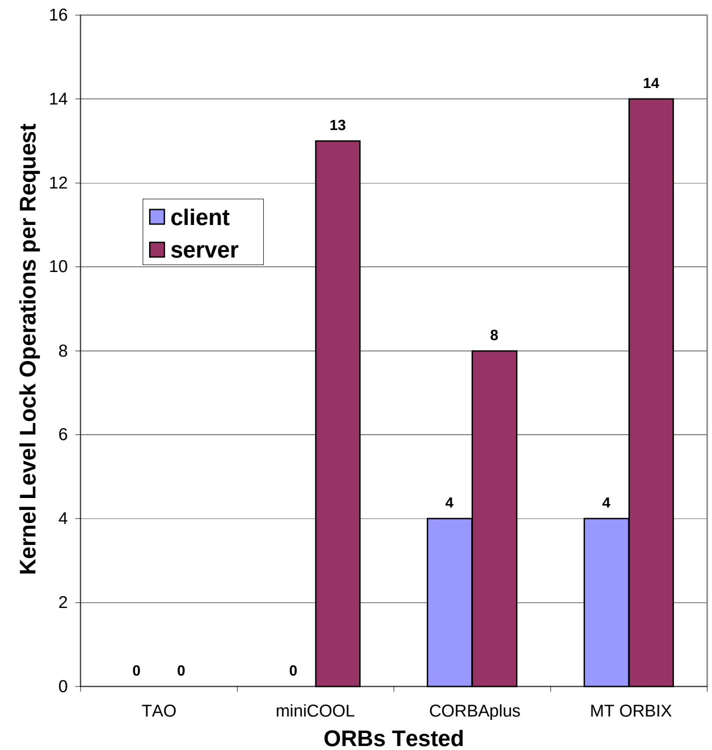
ORB Jitter Results



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

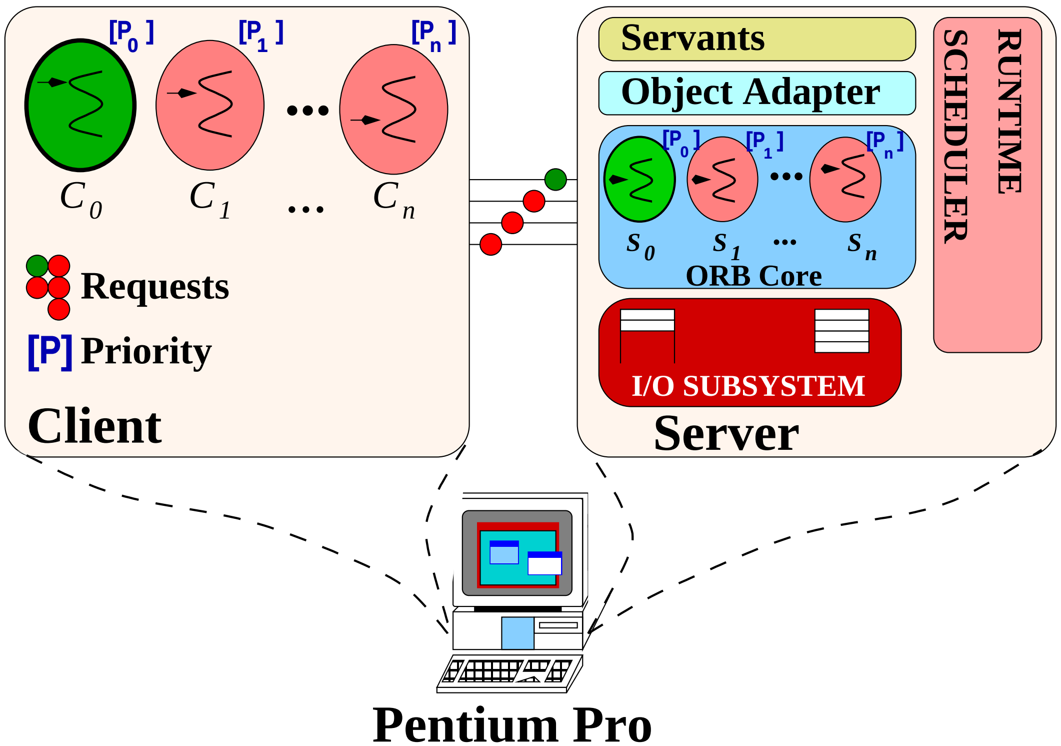
User-level and Kernel-level Locking Overhead





TAO is carefully designed to minimize memory allocation and locking

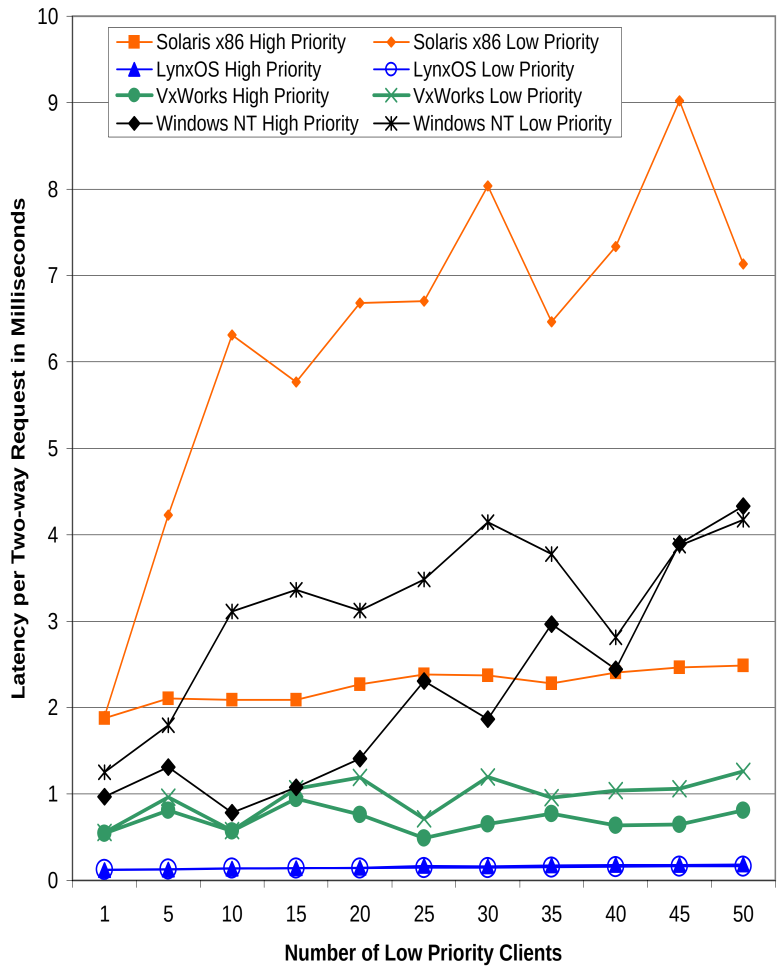
Real-time OS/ORB Performance Experiments



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

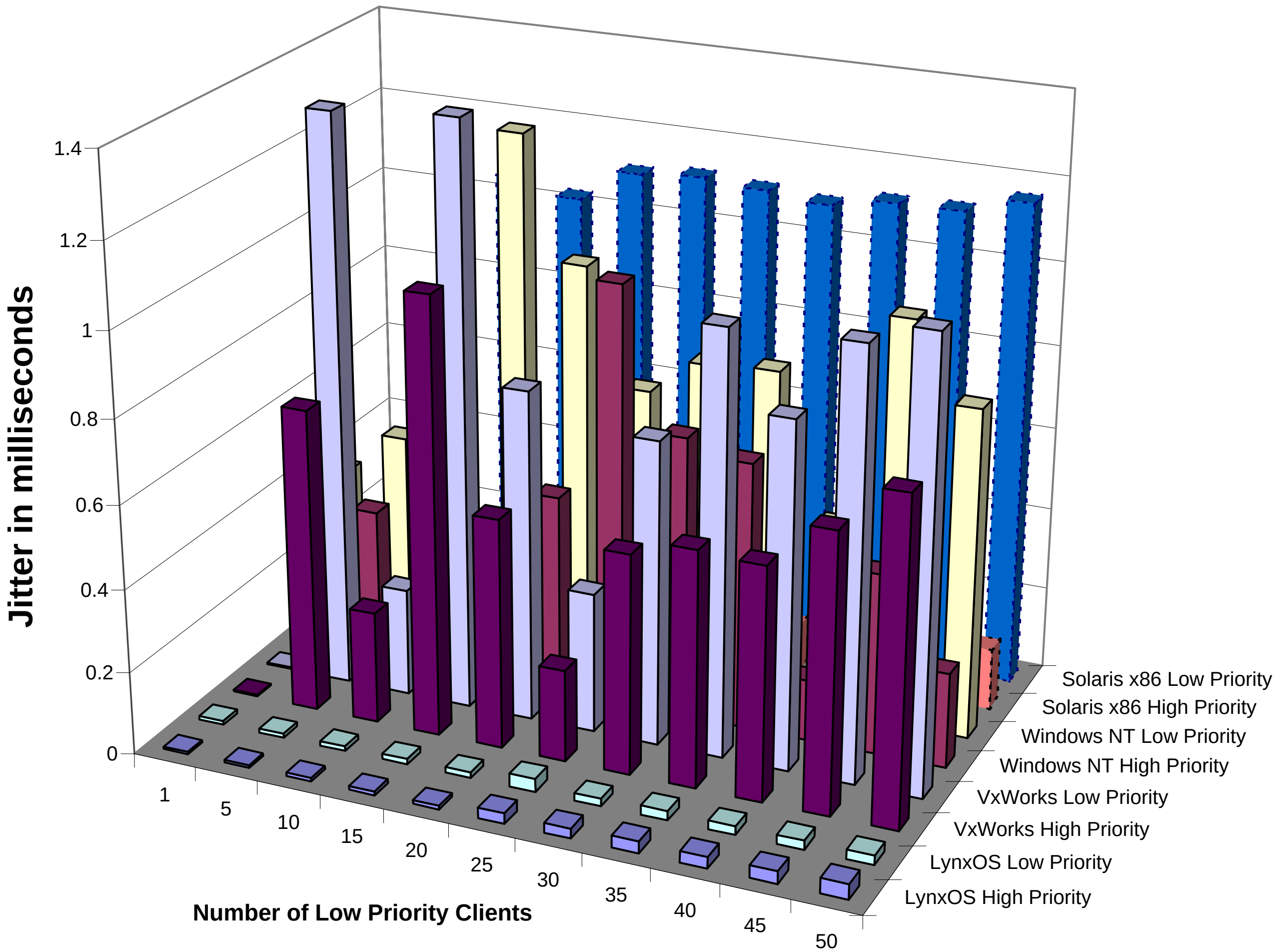
- Vary OS, hold ORBs constant
- Methodology
  - 1 high-priority client
  - *1..n* low-priority clients
  - Server uses *thread-per-priority*
    - *Highest* real-time priority for high-priority client
    - *Lowest* real-time priority for low-priority clients

Real-time OS/ORB Performance Results



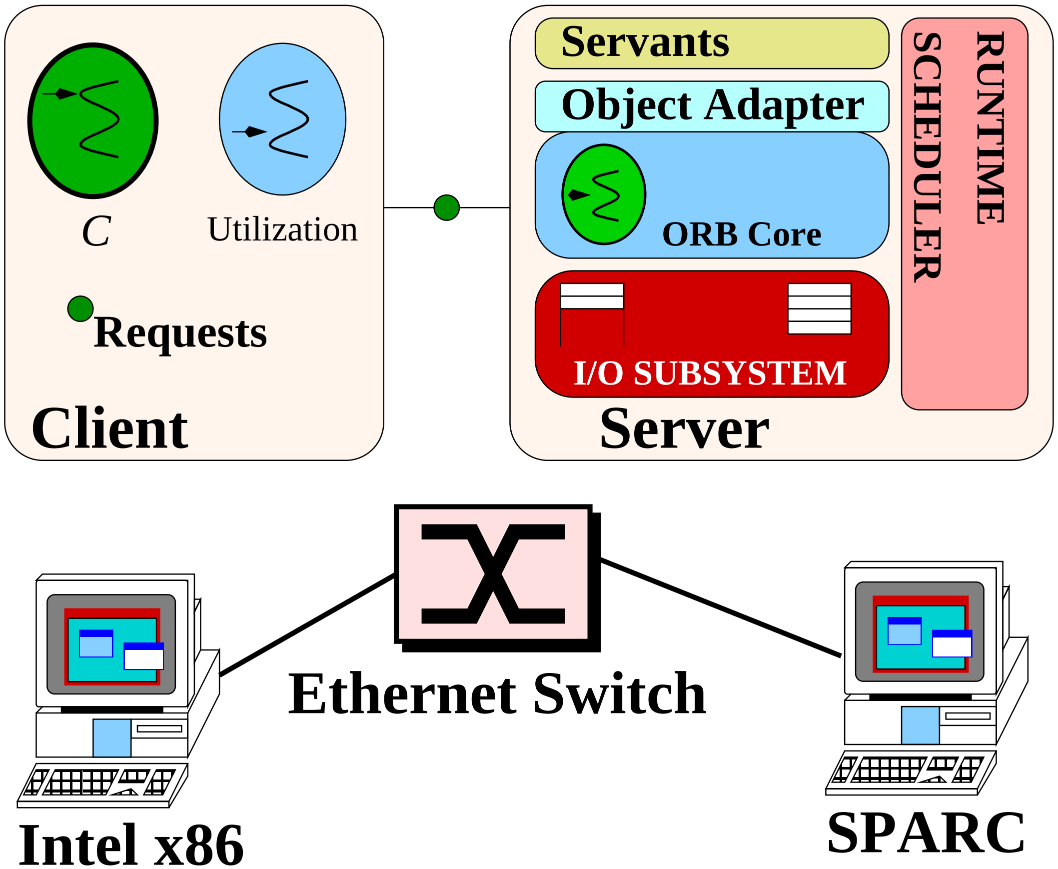
- **Synopsis of results**
  - RTOS's provide lowest latency
  - RTOS's minimize priority inversion
  - ORB (TAO) provides low latency and avoids priority inversion
    * *i.e.*, high priority client always has lowest latency

Real-time OS/ORB Jitter Results



- **Definition**
  - Standard deviation from average latency
- **Synopsis of results**
  - Some RTOS's provide low jitter
  - ORB (TAO) doesn't introduce jitter

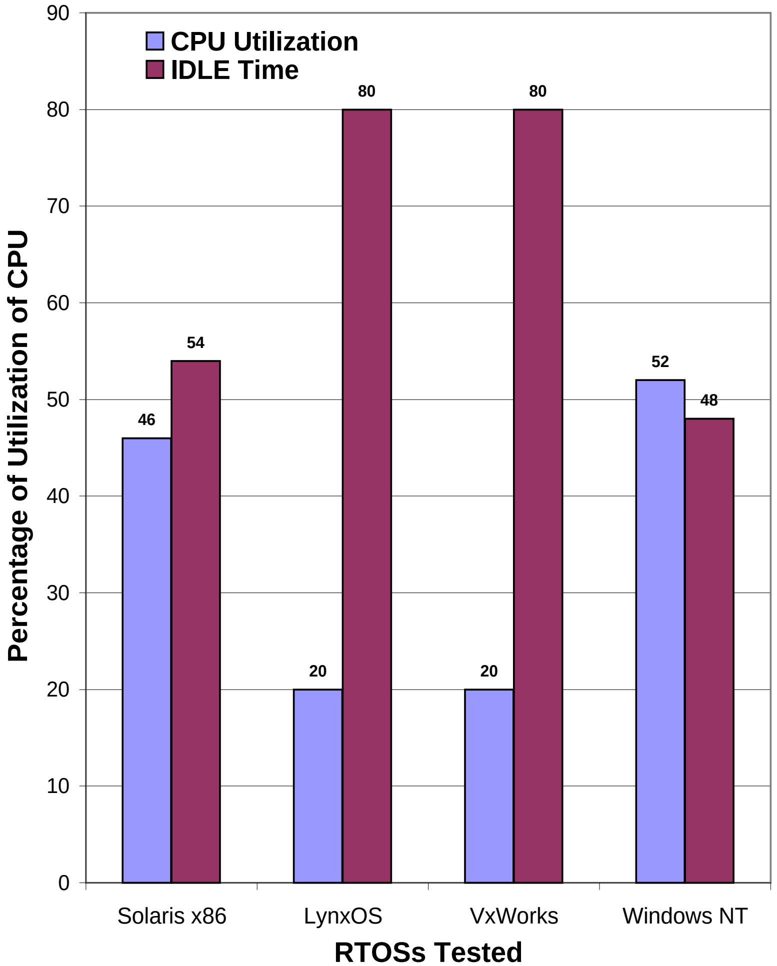
Real-time OS/ORB CPU Utilization Experiments



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

- Vary ORBs, hold OS constant
- Methodology
  - 1 client thread
  - 2 server threads
    - 1 thread services client
    - 1 thread factors prime numbers

Real-time OS/ORB CPU Utilization Results



- **Synopsis of results**
  - RTOS's provide highest effective utilization
  - ORB (TAO) processing uses ∼20% of the CPU

Concluding Remarks

- TAO is currently used at Boeing for avionics mission computing
  - Initial flight dates are mid-summer 1998
- Extensive benchmarks demonsrate it is possible to meet stringent performance goals with real-time CORBA
  - *e.g.*, for Boeing, target latency for CORBA oneway operations is 150 μsecs for 100 Mhz PowerPC running over MVME 177 boards
- Technology transfer to commercial vendors via OMG RT SIG and DARPA Quorom program