The Center for Distributed Object Computing

Research Synopsis

David L. Levine
www.cs.wustl.edu/∼levine/

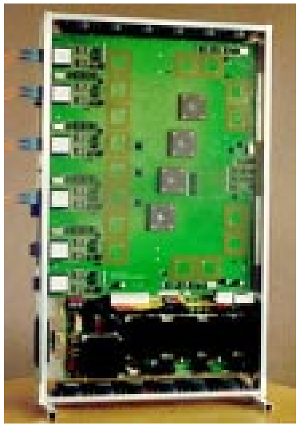
Director, Center for Distributed Object Computing
Computer Science Dept.
Washington University, St. Louis

Sponsors

NSF, DARPA, Bellcore/Telcordia, BBN, Boeing, CDI/GDIS, Comverse, Hughes, Kodak, Lockheed, Lucent, Microsoft, Motorola, Nokia, Nortel, OCI, OTI, Raytheon, SAIC, Siemens SCR, Siemens MED, Siemens ZT, Sprint, USENIX

24 November 1999

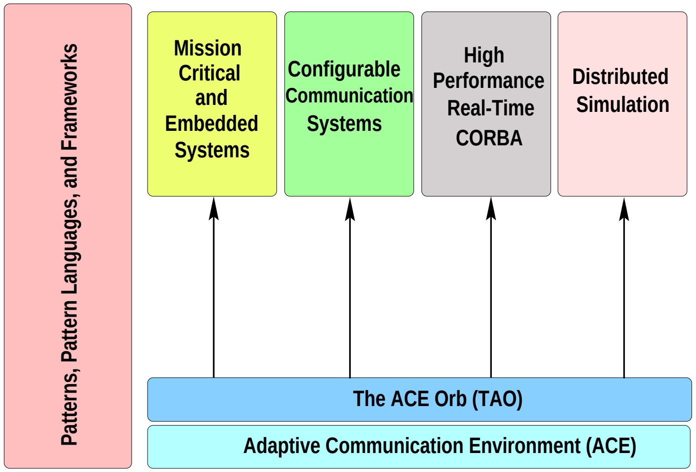
Motivation: the Communication Software Crisis



www.arl.wustl.edu/arl/

- **Symptoms**
  - Communication ***hardware*** gets smaller, faster, cheaper
  - Communication ***software*** gets larger, slower, more expensive
- **Culprits**
  - ***Inherent*** and ***accidental*** complexity
- **Solution Approach**
  - ***Standard communication middleware***

DOC Center Research Focus



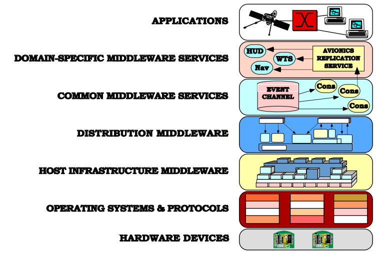
DOC Center Members

- David L. Levine, Director
- Fred Kuhns, Associate Director
- Douglas C. Schmidt, Former Director
- Full-time staff: C. Gill, C. O'Ryan, J. Parsons, I. Pyarali, N. Wang
- PhD students: C. Gill, J. Hu, C. O'Ryan, O. Othman, I. Pyarali, N. Wang
- Masters students: L. Baker, D. Brunsch, P. Gore, V. Kachroo, Y. Krishnamurthy, B. Natarajan, K. Parameswaran, J. Parsons, M. Spivak
- Undergrads: K. Pathayapura

DOC Center Sponsors

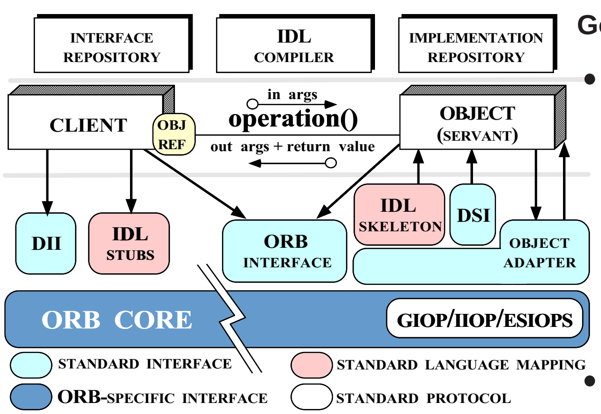
Bellcore/Telcordia	Motorola
BBN	Nokia
Boeing	Nortel
CDI/GDIS	NSF
Comverse	OCI
DARPA	OTI
Hughes	Raytheon
Kodak	SAIC
Lockheed Martin	Siemens
Lucent	Sprint
Microsoft	USENIX

Problem: Lack of QoS-enabled Middleware



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

Candidate Solution: CORBA

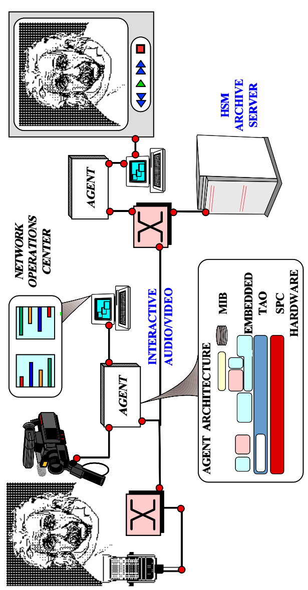


www.cs.wustl.edu/~schmidt/corba.html

Goals of CORBA

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

Caveat: Requirements/Limitations of CORBA for QoS-enabled Systems



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

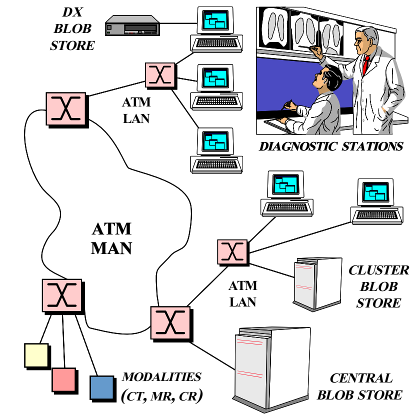
Requirements

- Location transparency
- Performance transparency
- Predictability transparency
- Reliability tranparency

Limitations

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

Problem: Optimizing Complex Software



www.cs.wustl.edu/∼schmidt/JSAC-99.ps.gz

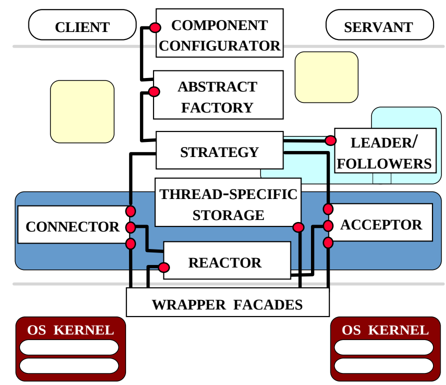
Common Problems →

- Optimizing complex software is hard
- Small "mistakes" can be costly

Solution Approach (Iterative) →

- Pinpoint overhead via *white-box* metrics
  - *e.g.*, `Quantify` and `VMEtro`
- Apply patterns and framework components
- Revalidate via white-box and black-box metrics

Solution 1: Patterns and Framework Components



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

Definitions

- *Pattern*
  - A solution to a problem in a context
- *Framework*
  - A "semi-complete" application built with components
- *Components*
  - Self-contained, "pluggable" ADTs

Solution 2: ORB Optimization Principle Patterns

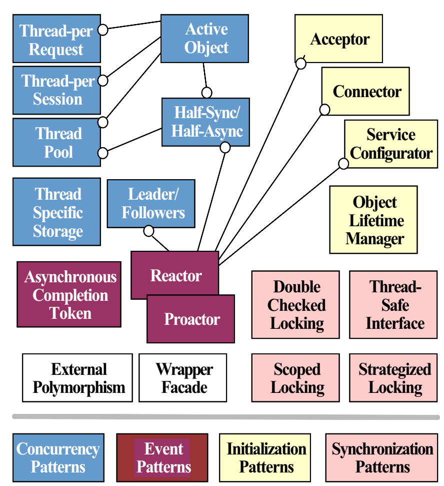
Definition

- ***Optimization principle patterns*** document rules for avoiding common design and implementation problems that can degrade the efficiency, scalability, and predictability of complex systems

Key Principle Patterns Used in TAO

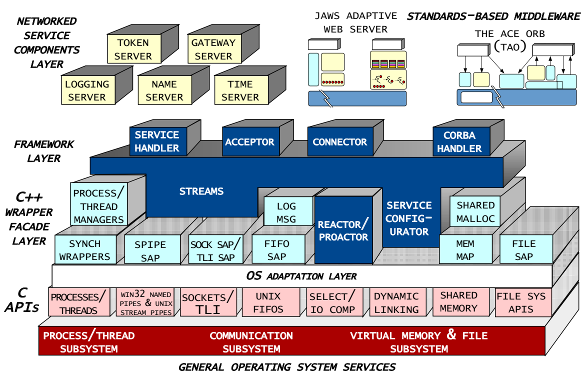
#	Principle Pattern
1	Optimize for the common case
2	Remove gratuitous waste
3	Replace inefficient general-purpose functions with efficient special-purpose ones
4	Shift computation in time, *e.g.*, precompute
5	Store redundant state to speed-up expensive operations
6	Pass hints between layers and components
7	Don't be tied to reference implementations/models
8	Use efficient/predictable data structures

Patterns for Communication Middleware



- **Observation**
  - *Failures rarely result from unknown scientific principles, but from failing to apply proven engineering practices and patterns*
- **Benefits of Patterns**
  - Facilitate design reuse
  - Preserve crucial design information
  - Guide design choices

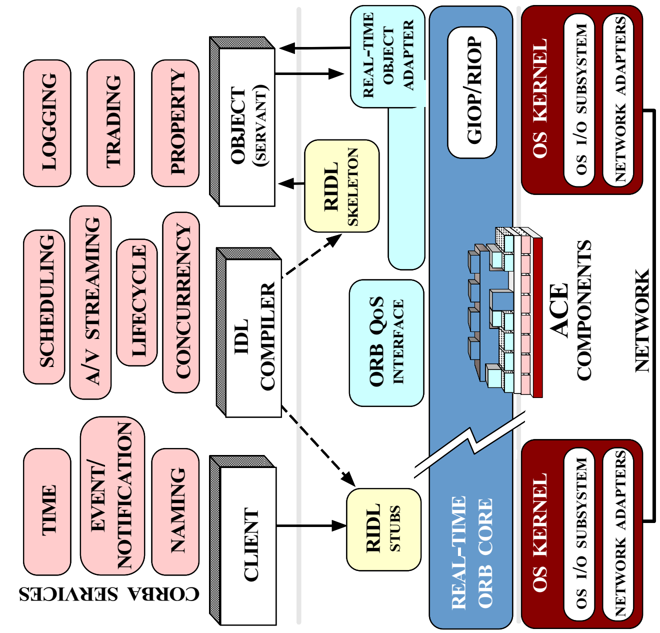
The ADAPTIVE Communication Environment (ACE)



- **ACE Overview**
  - Concurrent OO networking framework
  - Available for C++ and Java
  - Ported to POSIX, Win32, VxWorks, Chorus, PharLap TNT, *et al.*

www.cs.wustl.edu/∼schmidt/ACE.html

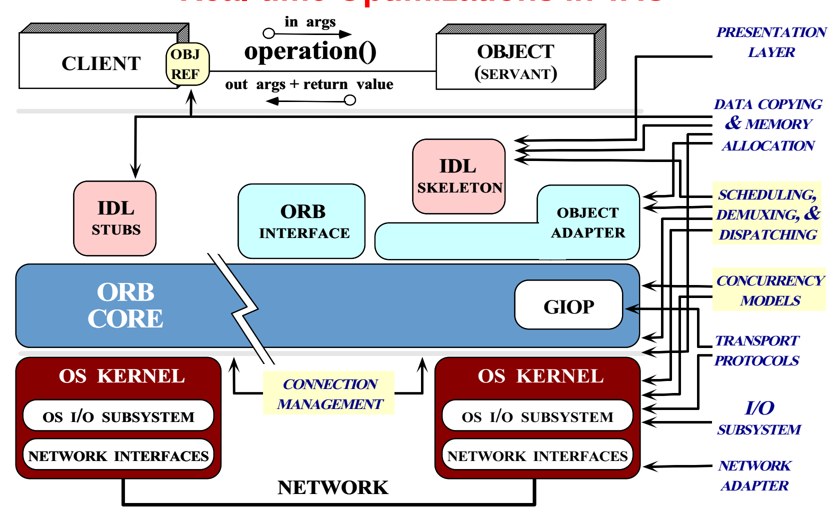
The ACE ORB (TAO) and Its CORBA Object Services



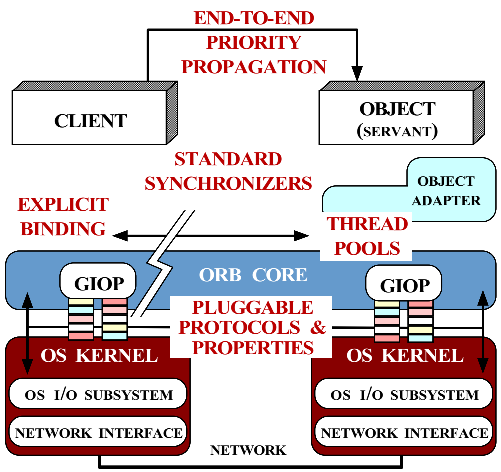
ACE and TAO Statistics

- Over 30 person-years of effort
  - ACE > 200,000 LOC
  - TAO > 125,000 LOC
  - TAO IDL compiler > 100,000 LOC
  - TAO CORBA Object Services > 150,000 LOC
- Ported to POSIX, Win32, VxWorks, *et al.*
- Large user community
  - www.cs.wustl.edu/∼schmidt/ACE-users.html
- Currently used by dozens of companies
  - Bellcore, Boeing, Ericsson, Kodak, Lockheed, Lucent, Motorola, Nokia, Nortel, Raytheon, SAIC, Siemens, etc.
- Supported commercially
  - ACE → www.riverace.com
  - TAO → www.theaceorb.com

Real-time Optimizations in TAO

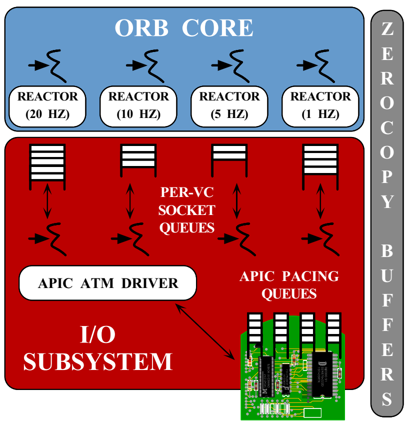


New TAO Features and Optimizations



- **New Features**
  - Real-time CORBA
  - Minimum CORBA
  - CORBA Messaging
  - Fault Tolerance
- **URL**
  - ~schmidt/TAO-status.html

Integrating TAO with ATM I/O Subsystem

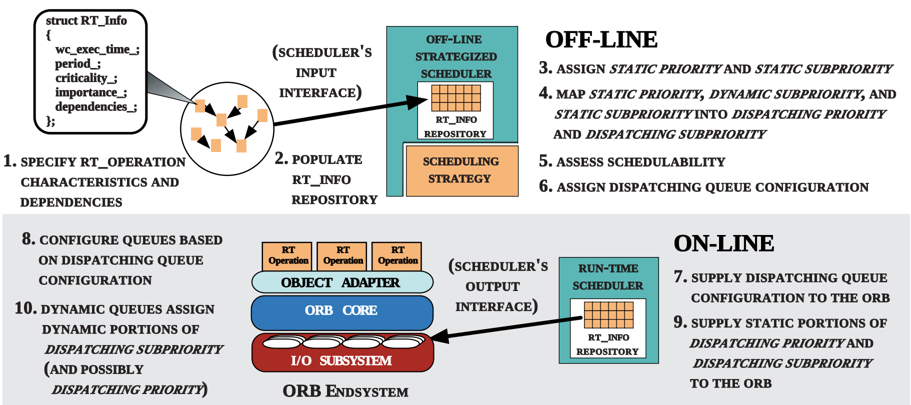


~schmidt/RIO.ps.gz

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

Strategized Scheduling Framework

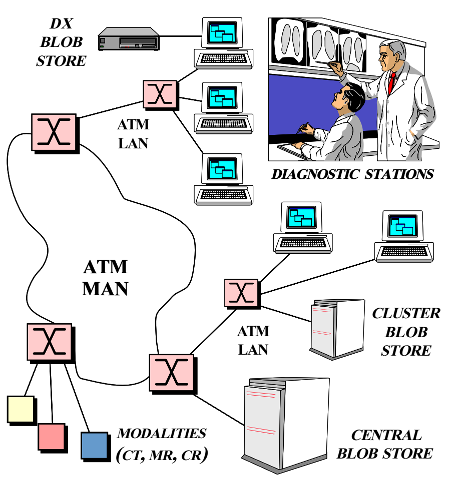


Use-cases for ACE and TAO

Domains

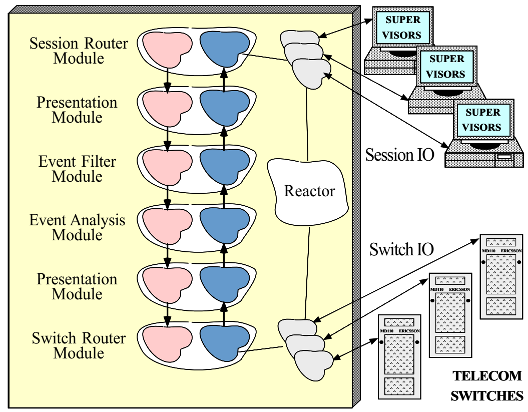
- Electronic medical imaging
- Network management
- Wireless personal communication systems (PCS)
- Real-time avionics mission comuting
- Multimedia services
- Distributed interactive simulation

Applying ACE and TAO to Medical Imaging



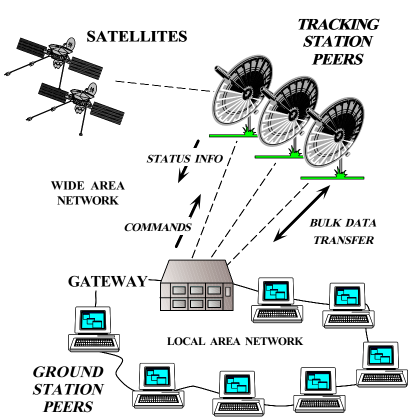
- **Domain Challenges**
  - Large volume of "Blob" data
    - ∗ *e.g.*, 10 to 40 Mbps
  - "Lossy compression" isn't viable
  - Prioritization of requests
- **URLs**
  - ∼schmidt/COOTS-96.ps.gz
  - ∼schmidt/av.ps.gz
  - ∼schmidt/NMVC.html

Applying ACE to Network Management



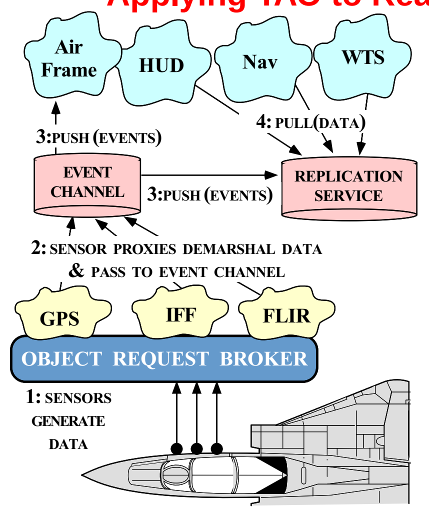
- **Domain Challenges**
  - Low latency
  - Multi-platform
  - Family of related services
- **URLs**
  - ∼schmidt/DSEJ-94.ps.gz
  - ∼schmidt/ECOOP-95.ps.gz

Applying ACE to Global PCS



- **Domain Challenges**
  - Long latency satellite links
  - High reliability
  - Prioritization
- **URL**
  - ∼schmidt/TAPOS-95.ps.gz

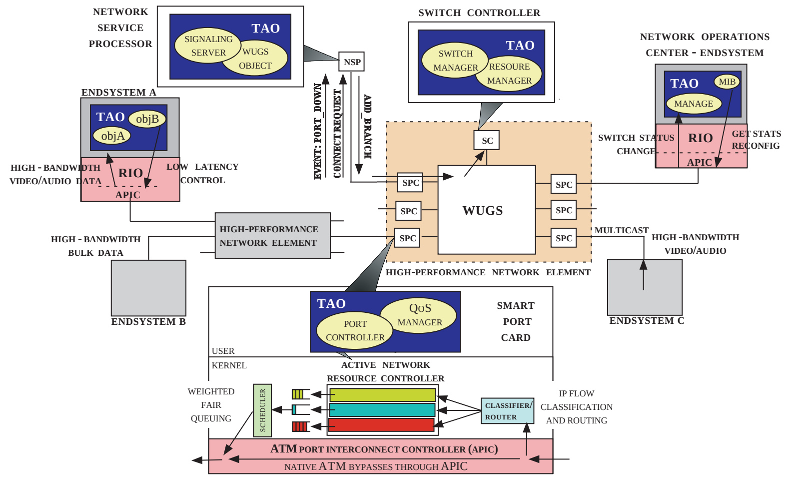
Applying TAO to Real-time Avionics



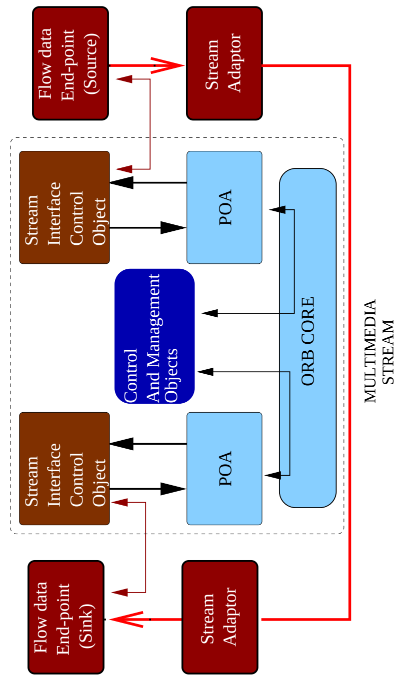
- **Domain Challenges**
  - Real-time periodic processing
  - Complex dependencies
  - Very low latency
- **URL**

  ∼levine/research/JSAC-98.ps.gz

Open ATM Signaling & Control



Audio/Video Streaming



Efficiency

- Sockets for data transfer to get high performance

Flexibility

- Uses CORBA for control messages and properties

Concluding Remarks

- Researchers and developers of distributed, real-time applications confront many common challenges
  - *e.g.*, service initialization and distribution, error handling, flow control, scheduling, event demultiplexing, concurrency control, persistence, fault tolerance
- Successful researchers and developers apply *patterns*, *frameworks*, and *components* to resolve these challenges
- Careful application of patterns can yield efficient, predictable, scalable, *and* flexible middleware
  - *i.e.*, middleware performance is largely an "implementation detail"
- Next-generation ORBs will be highly QoS-enabled, though many research challenges remain