Measuring OS Support for Real-time CORBA ORBs

Christopher D. Gill
Washington University, St. Louis
cdgill@cs.wustl.edu

27 January 1999

http://www.cs.wustl.edu/~cdgill/words-99.{pdf, ps}.gz

Sponsors: Boeing, CDI/GDIS, DARPA contract 9701516, Lucent, Motorola, NSF grant NCR-9628218, Siemens, and Sprint
Advantages of CORBA

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

www.cs.wustl.edu/~schmidt/ORB-endsystem.ps.gz
Limitations of CORBA for Real-time Systems

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

Diagram:

1. IDL STUBS
2. ORB INTERFACE
3. NETWORK
4. ORB CORE
5. GIOP/IIOP
6. OBJECT ADAPTER
7. IDL SKELETON
8. OBJECT (SERVANT)

Operations:

- in args: operation()
- out args + return value

Steps:

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
But Can It Perform?

- Low overhead
  - context switch time
  - request-response latency
  - ORB/endsystem CPU overhead

- Elimination of priority inversion
  - context switch time
  - request-response latency

- Predictability
  - request-response jitter

- Footprint
## Context Switch Time Measurements

<table>
<thead>
<tr>
<th>Operating System</th>
<th>Context Switch Time, (\mu\text{sec}) mean (standard deviation)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Suspend-Resume Test</td>
</tr>
<tr>
<td>VxWorks</td>
<td>0.946 (0.041)</td>
</tr>
<tr>
<td>LynxOS</td>
<td>N/A</td>
</tr>
<tr>
<td>Windows NT</td>
<td>1.41 (0.036)</td>
</tr>
<tr>
<td>Solaris</td>
<td>21.3 (0.569)</td>
</tr>
<tr>
<td>Linux</td>
<td>N/A</td>
</tr>
</tbody>
</table>

Table 1: Context Switch Time Measurements
Context Switch Time Measurements

- Measured two context switches for two-way CORBA operation, as expected
- Contribution of context switch time, 1 to 10 μsec, to latency is small
- Standard deviations are small, except on Solaris, where it may impair performance predictability
Priority Inversion Experiments

- One high-priority client
- 1..n low-priority clients
- Server factory implements *thread-per-connection*
  - Higher real-time priority for high-priority client
  - Lower real-time priority for all low-priority clients
- Each connection links client with its servant
**ORB Latency and Priority Inversion Results**

### Synopsis of results

- LynxOS and Linux provide lowest latency
- LynxOS and VxWorks do not exhibit priority inversion; non-RTOS's do
- Therefore, deterministic QoS support requires a RTOS
ORB Jitter Results

- **Definition**
  - Standard deviation from average latency

- **Synopsis of results**
  - LynxOS and VxWorks show low jitter
  - Linux, Windows NT, and Solaris show unacceptable jitter
Future Work

- Latency and jitter measurements
  - LynxOS offers low latency and jitter (high predictability)
  - VxWorks does also, with small number of low-priority clients; but does not scale well
  - Non-RTOS’ have high latency and/or jitter

- ORB/OS operation throughput
  - Fairly consistent results across platforms

- ORB/OS CPU processing overhead
  - Ranges from under 3 percent (Windows NT) to over 17 percent (VxWorks)
Concluding Remarks

- We can meet RT performance goals with CORBA, on RTOS’.

- Design and Implementation Challenges
  - Dynamic and hybrid scheduling of requests and operations
  - Specifying QoS requirements
  - Distributed QoS and integration with real-time I/O Subsystems
  - Alleviating priority inversion and non-determinism
  - Reducing latency/jitter for demultiplexing
  - Reducing presentation layer overhead
  - Optimizing IDL compilers
  - Maintaining small footprint
For Further Information

- **Performance Measurements:**
  - Demultiplexing latency:
    www.cs.wustl.edu/~schmidt/GLOBECOM-97.ps.gz
  - SII throughput:
    www.cs.wustl.edu/~schmidt/SIGCOMM-96.ps.gz
  - DII throughput:
    www.cs.wustl.edu/~schmidt/GLOBECOM-96.ps.gz
  - Latency, scalability:
    www.cs.wustl.edu/~schmidt/ICDCS-97.ps.gz
  - IIOP:
    www.cs.wustl.edu/~schmidt/IIOP.ps.gz
For Further Information

- **More detail on TAO:**
  www.cs.wustl.edu/~schmidt/RT-ORB.ps.gz

- **TAO Event Channel:**
  www.cs.wustl.edu/~levine/research/JSAC98.ps.gz

- **TAO static scheduling:**
  www.cs.wustl.edu/~schmidt/TAO.ps.gz

- **TAO dynamic scheduling:**

- **ORB Endsystem Architecture:**
  www.cs.wustl.edu/~schmidt/RT-middleware.ps.gz
For Further Information

- More detail on CORBA:
  www.cs.wustl.edu/~schmidt/corba.html

- ADAPTIVE Communication Environment (ACE):
  www.cs.wustl.edu/~schmidt/ACE.html

- The ACE ORB (TAO):
  www.cs.wustl.edu/~schmidt/TAO.html

- These slides:
  www.cs.wustl.edu/~cdgill/words-99.{pdf, ps}.gz