Real-Time Platforms

- Real-Time OS: Linux
- Real-Time Middleware: TAO
  - Event service
  - Single-processor scheduling
  - End-to-end scheduling
  - Aperiodic scheduling
- Real-Time Virtualization: RT-Xen
- Real-Time Cloud: RT-OpenStack
- Real-Time Parallel Computing: RT-OpenMP
Problems with Current Approaches

We increasingly rely on distributed real-time & embedded (DRE) systems

Characteristics of DRE Systems
- Network-centric
- Stringent quality of service (QoS) demands
- Resource constrained
- Mission-critical control

Technical limitations with today’s DRE infrastructure
- Stovepiped
- Proprietary
- Brittle & non-adaptive
- Expensive
- Vulnerable
Develop & apply the new generation of distributed object computing (DOC) middleware technologies to
1. **simultaneously control** multiple QoS properties &
2. **improve software** development quality, productivity, adaptability, & assurability

**Benefits of Middleware**
- Highly scalable QoS
- Enable new resource management capabilities
- Support common & open technology bases
- Leverage & enhance advances in assurance & security
Why Are We Succeeding

The past decade has yielded significant progress in QoS-enabled middleware, stemming in large part from the following trends:

- **Years of iteration, refinement, & successful use**
  - Real-time CCM
  - CORBA Component Model (CCM)
  - Component Models (EJB)
  - Real-time CORBA
  - CORBA & DCOM
  - DCE
  - Micro-kernels
  - RPC
  - ARPAnet

- **The maturation of middleware standards**
  - Applications
  - Domain-Specific Services
  - Common Services
  - Distribution Middleware
  - Host Infrastructure Middleware
  - Operating Systems & Protocols
  - Hardware

- **The maturation of component middleware frameworks & patterns**
  - NET, J2EE, CCM
  - Real-time CORBA
  - Real-time Java
  - SOAP & Web Services

---

<table>
<thead>
<tr>
<th>Year</th>
<th>1970</th>
<th>2005</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARPAnet</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Application Example: Avionics

Goals
• Apply COTS & open systems to mission-critical real-time avionics

Key System Characteristics
• Deterministic & statistical deadlines
  • ~20 Hz
• Low latency & jitter
  • ~250 us
• Periodic & aperiodic processing
• Complex dependencies
• Continuous platform upgrades

Key Results
• Test flown at China Lake NAWS by Boeing OSAT II '98, funded by OS-JTF
  • www.cs.wustl.edu/~schmidt/TAO-boeing.html
• Also used on SOFIA project by Raytheon
  • sofia.arc.nasa.gov
• First use of Real-time CORBA in avionics
• Drove Real-time CORBA standardization
Application Example: Hot Rolling Mills

Goals
• Control the processing of molten steel moving through a hot rolling mill in real-time

System Characteristics
• Hard real-time process automation requirements
  • *i.e.*, 250 ms real-time cycles
• System acquires values representing plant’s current state, tracks material flow, calculates new settings for the rolls & devices, & submits new settings back to plant

Key Software Solution Characteristics
• Affordable, flexible, & COTS
• Product-line architecture
• Design guided by patterns & frameworks

www.siroll.de
Application Example: Image Processing

Goals
- Examine glass bottles for defects in real-time

System Characteristics
- Process 20 bottles/sec
  - ~50 ms per bottle
- Networked configuration
  - ~10 cameras

Key Software Solution Characteristics
- Affordable, flexible, & COTS
- Embedded Linux (Lem)
- Compact PCI bus + Celeron processors
- Remote booted by DHCP/TFTP
- Real-time CORBA
CORBA
Common Object Request Broker Architecture

- CORBA specifications
  - OMG is the standards body
  - Over 800 companies
  - CORBA defines *interfaces*, not *implementations*

- Object Request Brokers (ORB) allow clients to invoke operations on distributed objects transparently of
  - Object location
  - Programming language
  - Operating system
  - Communication protocols and interconnect
  - Hardware
Client invokes operations on objects.

An Object =
  - An interface specified by an Interface Definition Language (IDL)
  - Servant(s) that implements the IDL interface
Stubs and Skeletons

- Translate between platform-dependent data formats and common data representation.
- Generated by an IDL compiler based on the IDL interface.
- Ensure platform/language transparency.
ORB Core

- Deliver requests to objects and responses to clients
- Communicate using a General Inter-ORB Protocol (GIOP)
  - e.g., Internet Inter-ORB Protocol (IIOP) on TCP
- Typically a run-time library linked to applications
Object Adapter

- Demultiplexes each incoming request to the right servant/operation
- Make the upcall to the operation
Limitations of CORBA

- Lacks QoS specification interfaces to applications
  - Applications cannot specify rate, deadline or importance

- Lacks QoS enforcement
  - Does not map task QoS specification to priorities of threads
  - Contains significant priority inversion

- Lacks performance optimization
  - Poor worst-case and average latency
Latencies and Priority Inversions

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
The ACE ORB (TAO)

• Open-source Real-Time CORBA
• >> 1M SLOC
• 100+ person years of effort
• Pioneered R&D on DRE middleware design & optimization

Objective
- Simplify the development of distributed real-time & embedded (DRE) systems

Approach
- Use standard technology, patterns, & frameworks

TAO Overview

- Based on ACE wrapper facades & frameworks
- Available on Unix, Win32, MVS, QNX, VxWorks, LynxOS, VMS, etc.
- Thousands of users around the world

- Commercially supported by many companies
  - OCI (www.theaceorb.com)
  - PrismTech (www.prismtechnologies.com)
  - more coming soon...
  - www.cs.wustl.edu/~schmidt/commercial-support.html
I/O Subsystem: Priority Inversions

- Messages are processed in FIFO order regardless of priorities

- Kernel has lower priority than real-time priorities
  - Processing of a high priority message in the kernel can be blocked by a lower priority task at the application level
I/O Subsystem: Solutions

- Early demultiplexing
- Prioritized kernel processing
- Map task priority to kernel thread
- Note: This needs kernel support
ORB Core: Priority Inversions

- Communication of different tasks shares a same socket connection.
- Incoming requests are demultiplexed to threads in FIFO order.
Server sets model: Each priority is processed by a separate thread.

A separate connection is maintained for each priority in the server ORB.
- Use buffered connections to reduce run-time overhead.

Suitable for fixed priority scheduling.
Layered demultiplexing is inefficient in terms of
- average latency
- worst-case latency
Object Adapter: Solutions

- Perfect hashing
  - Generate a hash function offline
  - Computational complexity $O(1)$
- De-layered active demultiplexing
  - Clients obtain index to (servant, operation) ahead of time
Reduce Priority Inversion

Conventional ORB

TAO
The Evolution of TAO

Dynamic/Static Scheduling

A/V Streaming

Static Scheduling (1.0)
- Rate monotonic analysis

Dynamic Scheduling (1.2)
- Earliest deadline first
- Minimum laxity first
- Maximal urgency first

Hybrid Dynamic/Static
- Demo in WSOA
- Kokyu integrated

A/V Streaming Service
- QoS mapping
- QoS monitoring
- QoS adaptation

ACE QoS API (AQoSA)
- GQoS/RAPI & DiffServ
- IntServ integrated with A/V Streaming & QuO
- DiffServ integrated with ORB
The Evolution of TAO

DYNAMIC/STATIC SCHEDULING

FT-CORBA & LOAD BALANCING

SECURITY

Real-time CORBA 1.0

A/V STREAMING

FT-CORBA (DOORS)
• Entity redundancy
• Multiple models
  • Cold passive
  • Warm passive
• IOGR
• HA/FT integrated.

Load Balancing
• Static & dynamic
• Integrated in TAO 1.3
• De-centralized LB
• OMG LB specification

SSL Support
• Integrity
• Confidentiality
• Authentication (limited)

Security Service (CSIv2)
• Authentication
• Access control
• Non-repudiation
• Audit
The Evolution of TAO

- Dynamic/Static Scheduling
- FT-CORBA & Load Balancing
- Notifications
- Transactions
- Security
- Audio/Video Streaming

Notification Service
- Structured events
- Event filtering
- QoS properties
  - Priority
  - Expiry times
  - Order policy
- Compatible w/Events

Real-time Notification Service

Object Transaction Service
- Encapsulates RDBMs
- www.xots.org

Real-time CORBA 1.0

IDL Stubs

QoS Interface

IDL Skeletons

Portable Object Adapter

Real-time CORBA Core

GIOP/IIOP/ESIOPs

IOP

In args

operation()

out args + return

Server

Client

www.xots.org

The Evolution of TAO
Acknowledgement

Slides partially borrowed from Doug Schmidt’s Real-Time CORBA tutorial.