SYSTEM SUPPORT FOR IMPLEMENTATION AND EVALUATION OF REAL-TIME ORBS

Douglas Niehaus
Information and Telecommunication Technology Center
University of Kansas
Lawrence, KS
niehaus@ittc.ukans.edu

Spartan Conference
MOTIVATION

- CORBA is an attractive implementation technology for emerging uses of the next generation telecommunication networks
- Applications with real-time constraints are increasing in number and importance
- Real-time applications require the ability to provide performance guarantees
  - Guarantees are based on predictability of behavior
  - Behavior of an application is influenced by all aspects of the system
  - A system is only as predictable as its least predictable component
- Detailed performance evaluation of ORB implementations and operational scenarios is necessary to achieve performance and predictability goals
- Explicit system support for real-time applications is required to meet the requirements of real-time applications
System Support for Implementation and Evaluation of Real-Time ORBs

RELATED KU PROJECTS

- **KU Real-Time (KURT)**
  - Modifications to Linux to increase time resolution and support for *firm real-time* applications

- **Performance Measurement Object (PMO)**
  - NetSpec based test demon explicitly addressing the performance evaluation of ORBs in general and real-time ORBs in particular

- **Data Stream Kernel Interface (DSKI)**
  - Pseudo-device driver providing an interface for specifying and collecting a time-stamped kernel event stream

- **Smart-GDB**
  - Extension to GDB to support user programmable custom debugging interfaces → thread aware debugging
MODIFICATIONS TO LINUX FOR REAL-TIME

• Provides support for firm real-time applications
  – Those with stringent but not individually critical timing constraints
  – Multi-media, telecommunications, interactive virtual environments
• Students: Balaji Srinivasan, Robert Hill, Shyam Pather, Furquan Ansari, Raghavan Menon, Jason Keimig, Apurva Sheth

• Increased temporal resolution (UTIME)
  – Generic Linux has a timing resolution of 10ms, similar to most commercial systems
  – Modified Linux’s time keeping method to provide microsecond temporal resolution with minimal increase in overhead
  – Added the ability to schedule events with a resolution of 10 to 20 microseconds
  – Good for general use as well as real-time support
• KU Real-Time (KURT)
  – Added system calls and modified scheduling to support *explicit time line* scheduling
    * Schedule explicitly states when each real-time process will run
    * Periodic processes supported by executing a time line cyclicly
  – Real-time Modes
    * Focussed: executes *only* the designated real-time processes
    * Integrated: executes non-real-time processes as resources are available

• KURT’s predictability and temporal resolution significantly exceeds that of many applications using specialized and expensive commercial real-time OSs

• More Information:
  – http://hegel.ittc.ukans.edu/projects/utime
  – http://hegel.ittc.ukans.edu/projects/kurt
System Support for Implementation and Evaluation of Real-Time ORBs

UTIME CLOCK DRIFT

Adjusted UTIME

Standard Linux and UTIME with XNTP

UTIME

Standard Linux

Douglas Niehaus

Information and Telecommunication Technology Center

University of Kansas
System Support for Implementation and Evaluation of Real-Time ORBs

LINUX FIFO SCHEDULING PERFORMANCE

KURT scheduler

- - - - 1 process
- - - - 4 processes
- - - - 8 processes
- - - - 10 processes

% of events

Time difference between consecutive loops (microseconds) x 10^4

Douglas Niehaus
Information and Telecommunication Technology Center
University of Kansas
PERFORMANCE MEASUREMENT OBJECT

- NetSpec test demon which is also an ORB object and thus provides the ability to implement ORB based performance evaluation benchmarks as sets of NetSpec scripts
- Initial development and testing of the framework is complete
- Extension of the framework and implementation of comprehensive benchmarks is the next step
- Students: Anil Gopinath, Sridhar Nimmagadda, Chanaka Liyanaarachchi
- More Information:
  - http://hegel.ittc.ukans.edu/projects/pmo
PMO ARCHITECTURE

- NetSpec controls the experiment as directed by the script
- Conventional NetSpec test and measurement (DSKI) demons can also be used
#include "corba-requestresp.h"

cluster
{
    corba RECEIVING_HOST
    {
        type = TYPE_OF_TEST (orb = ORB_NAME,
            mode = receiver,
            numsamples = NUMSAMPLES,
            minsize = MINSIZE,
            maxsize = MAXSIZE,
            multiples = MULTIPLES,
            predelay = PREDELAY,
            postdelay = POSTDELAY,
            duration = DURATION
        );
        protocol = PROTOCOL_NAME;
        criteria = CRITERIA;
        qos = QOS;
        objname = obj1;
        own = SENDING_HOST (interface = INTERFACE);
    }

corba SENDING_HOST
{
    type = TYPE_OF_TEST (orb = ORB_NAME,
        mode = sender,
        numsamples = NUMSAMPLES,
        minsize = MINSIZE,
        maxsize = MAXSIZE,
        multiples = MULTIPLES,
        predelay = PREDELAY,
        postdelay = POSTDELAY,
        duration = DURATION
    );
    protocol = PROTOCOL_NAME;
    criteria = CRITERIA;
    qos = QOS;
    objname = obj2;
    own = RECEIVING_HOST (interface = INTERFACE,
        port = PORT);
}
}
LATENCY BENCHMARK TEST

- Latency observed by a sender object while transferring a sequence of CORBA data types to a receiver object when they are located on different machines

![Graph showing latency benchmark test results for different data types. The x-axis represents the length of the sequence, and the y-axis represents latency in microseconds. Different data types such as double, octet, short, string, and long are represented with distinct markers. The graph indicates a linear relationship between the length of the sequence and latency.]
REAL-TIME SCHEDULING SERVICE

- ORB service providing an interface to the underlying real-time services
- KURT provides scheduling and execution support, being extended to include ATM QoS
## RTSS SENDER LATENCY EXPERIMENT

<table>
<thead>
<tr>
<th>Scheduling Policy</th>
<th>Startup latency (millisecs)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCHED_NORMAL</td>
<td>19.29</td>
</tr>
<tr>
<td>SCHED_FIFO</td>
<td>16.00</td>
</tr>
<tr>
<td>SCHED_KURT</td>
<td>15.87</td>
</tr>
</tbody>
</table>

![Bar chart showing latency for different scheduling policies](chart.png)

The bar chart above illustrates the latency in milliseconds for different lengths of the CORBA string sequence. The x-axis represents the length of the CORBA string sequence, while the y-axis shows the latency in milliseconds. The chart compares the performance of SCHED_OTHER, SCHED_FIFO, and SCHED_KURT scheduling policies.
CONCLUSIONS

- Real-time applications under CORBA will continue to increase in number and importance
- Performance of real-time systems is influenced by all aspects of the system requiring detailed benchmarking and specific support for real-time
- KURT support for real-time and PMO support for ORB performance evaluation provide important elements of the total support which will be required to successfully deploy real-time ORB based applications
FUTURE WORK

- Improve KURT predictability and performance by adapting disk and other I/O subsystems to real-time support
- Adapt KURT to provide specific support to Real-Time Scheduling Service (RTSS) in general and merge with The Ace ORB (TAO) RTSS in particular
- Create ORB specific support in by adding integration between the I/O and scheduling subsystems to KURT
- Extend PMO based benchmarks to first cover traditional point-to-point throughput and latency tests, and then a wide range of application scenarios
- Extend PMO benchmarks specific to real-time
- Fully automate PMO and Netspec based ORB benchmarks and analysis