

Real-time CORBA on MICO-MT – Design, Implementation, Performance and Application

A. David Selvakumar, P.M.Sobha
Real-time Systems Group
Centre for Development of Advanced Computing (C-DAC)
No. 1, Old Madras Road, Bangalore – 560038, India
Email: david, sobha@cdacb.ernet.in

R. Pitchiah
Department of Information Tech.,
Electronics Niketan,
New Delhi, India
pitchiah@mit.gov.in

Abstract

*Mission critical systems like avionics, process control, telecommunication infrastructure etc with distributed heterogeneous environment demand the underlying middleware, OS and networks for interfaces to enhance the predictability, dependability and scalability of the system. The Object Management Group (OMG) has addressed middleware level real-time and fault tolerance issues in Real-Time CORBA (RT-CORBA) [1] and Fault-Tolerant CORBA standards respectively. This paper first describes the design and implementation of real-time extensions on multithreaded MICO (MICO-MT) [2]. Second, the paper describes the design of a generic schedulability analysis tool and our implementation of static real-time CORBA scheduling service. Finally, the paper presents performance measurements of real-time MICO-MT ORB on QNX real-time platform (RTP) and application of the implementation for our Supervisory Control And Data Acquisition System (SCADA).
Key words: Real time CORBA, schedulability analysis, static scheduling service, SCADA, DASPCP.*

1. Introduction

Conventional CORBA Object Request Broker (ORB) implementation exhibits substantial priority inversion and non-determinism, which makes them unsuitable for applications with stringent real-time requirements [3]. Moreover, conventional ORBs do not provide interfaces for applications to specify QoS requirements. OMG specification on RT-CORBA defines standards based end-to-end QoS support at multiple levels in distributed real-time systems. RT-

CORBA finds application in distributed systems, which need prioritized invocations, predictable response time and control of system resources from application level.

There have been many research efforts on RT-CORBA. The ACE TAO ORB is a C++ ORB with most of the features and services defined in the CORBA 2.6 specification, which includes the RT-CORBA specification. In TAO, only the minimum schedulability criterion, Liu-Layland theorem [10], is checked for ensuring schedulability [4] which is a necessary but not sufficient condition to ensure schedulability of the system [6] [10]. Moreover the scheduling service is currently integrated with the TAO's real-time Event Service [5] and not the ORB core. It doesn't address the mechanisms like Distributed Priority Ceiling Protocol [6], Distributed Priority Inheritance Protocol [7] to be adapted to avoid distributed system priority inversion problem.

ORBExpressRT [7] is a RT-CORBA compliant implementation which does not provide any tool for schedulability analysis and no support for scheduling service features. The latest release of HighlanderComm's VisiBroker™-RT does not address schedulability analysis, scheduling service and mechanism to avoid distributed system priority inversion.

A dynamic real-time CORBA system has been developed under the combined effort of the University of Rhode Island and the SPAWAR Systems Center. The architecture provides best-effort end-to-end enforcement of soft real-time client method requests. It is based on RapidSched for schedulability analysis. Rapid Sched [8] is an implementation of the RT-CORBA Scheduling Service and uses the Rapid RMA tool to generate the scheduling parameters and

automatically set them in RT-CORBA application code. The system does not support QNX RTP and MICO-MT ORB.

2. MICO-MT Architecture

MICO-MT is an Open Source, CORBA 2.3 compliant, multi-threaded Portable Object Adapter (POA) based ORB.

A multi-threaded ORB executes up-calls concurrently. Each incoming request gets assigned a new thread. Invocation results are returned to the ORB asynchronously. MICO-MT uses thread-per-request model. Reader/Writer threadpools are used for reading/writing to the I/O subsystem. The Root POA receives requests from ORB core for all POAs, inspects the target object references and dispatches the request to concerned child POAs if required, or executes the method if the object is part of Root POA. MICO-MT based server concurrently services multiple clients and each connection has a dedicated pair of read/write threads. Accept thread listens for new connections. MICO-MT does not support interfaces/architecture specified by RT-CORBA standard like POA level threadpool/threadpool lane, priority banded multiplexed/non-multiplexed connections etc. Each request is serviced by decoder threadpool as thread per request model. MICO-MT's ORB core is thread-safe and client can send invocations on multiple threads. Client can set the priority of the thread using the OS interface. But the MICO-MT ORB lacks mechanism to preserve the priority across the invocation path to avoid priority inversion.

3. The Real-time MICO-MT ORB

Figure 1 depicts MICO-MT with RT-CORBA extensions. RT-CORBA:: Current object facilitates an application to specify the priority attribute for an invocation. The invocation priority is preserved as thread specific data of the thread, which performs the invocation. The client ORB analyses the Interoperable Object Reference (IOR) of the object and retrieves the priority model (i.e.) client propagated or server declared. Accordingly it sets the priority for the invocation. The protocol supported by server is utilized to select the communication mechanism. The priority information is used by ORB to dispatch the request through the appropriate priority banded connections.

Each connection is associated with a single priority to prevent priority inversion from end to end, which might occur if the same connection is used for more than one priority level.

The server ORB's I/O subsystem's reader thread, receives the request and dispatches it to the ORB core for decoding as GIOPConnMsg message. The decoder thread builds ORBPOAMsg message and sends it to the invocation adapter, the Root POA. The root POA dispatches the method if the object is part of Root POA or sends an InterPOAMsg message to the respective child POA. The ORBPOAMsg message is processed by a thread pool of RootPOA. InterPOAMsg message is processed by child POA thread pool. All the threads processing the message has a priority which is either the priority specified by the client or declared by the server. The reply is send back by writer thread through the same connection.

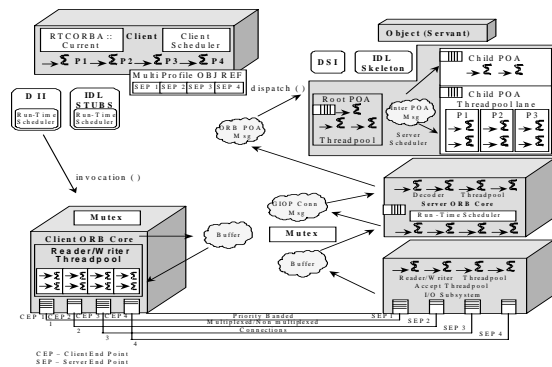


Figure1.Real-time MICO-MT ORB

The ORB has been enhanced to support explicit and implicit bind to server by RT-CORBA clients as given in the RT-CORBA specification. Dedicated non - multiplexed connections are feasible. The ORB also provides an interface to specify timeout on an invocation to ensure predictability.

4. Schedulability Analyzer

The schedulability analyzer is the tool that analyses the schedulability of a given set of periodic tasks on a single processor node or a set of single processor nodes (distributed system). The resource allocation policies supported are First In First Out (FIFO), Priority Inheritance Protocol (PIP), and Priority Ceiling Protocol (PCP) for single node analysis and Distributed Priority Ceiling Protocol

(DPCP) / Distributed affected Set Priority Ceiling Protocol (DASPCP) for distributed systems. It performs Rate Monotonic Analysis (RMA) [9] for an independent task set with deadlines less than or equal to period as well as arbitrary deadlines. The analyzer, after assessing the schedulability of the system, assigns unique priorities to each task. DPCP or DASPCP is chosen for analyzing CORBA based distributed system based on concurrency level (i.e.) object or method. It assumes static scheduling of tasks. All resources (CORBA servers or objects or method of an object) within the distributed system are assumed to be having the same allocation policies.

The schedulability analyzer allocates priority to all the tasks as monotonically decreasing function of period, computes the priority ceiling values for shared global / local critical sections (CORBA objects) and estimates the blocking time for all tasks as per the resources allocation policy. The analyzer supports CORBA model and the assigned parameters can be exported to all nodes of the distributed system. The system task set is analyzed for schedulability based on the following with the specified resource access policy:

- Liu-Layland theorem [10] with and without blocking period (resource access time) for minimum schedulability criteria.
- Calculating Response time for each task with and without blocking period.
- Test very scheduling points based on Lehoczky, Sha and Ding algorithm [11] with and without blocking period.

5. RT-CORBA Scheduling Service

RT-CORBA clients and servers can be on different nodes of a distributed system and scheduling tasks in a distributed system involves assigning priority to tasks and controlling access to shared resources, the servers.

Schedulability analysis of a distributed system is complicated due to the fact that tasks can access the resources residing on remote nodes as well as local nodes. The DPCP extends the basic Priority Ceiling Protocol (PCP) for distributed system. DASPCP [12] considers each method of an object as a resource. The schedulability analysis using DASPCP accounts for remote resources accesses. We use this protocol to model RT-CORBA applications where RT-CORBA servers/objects/methods are considered as remote resources for RT-CORBA clients.

The RM scheme assigns global priorities to each task and these global priorities are mapped to OS scheduler level local priorities. The tasks execute at local priorities. To avoid priority inversion due to multiple global priorities mapping into single local priority, the model assumes a restricted range of global priorities. Global priorities and local priorities are one-to-one mapped.

Figure 1 shows the run-time scheduling service components. The run-time scheduler part of server ORB core enforces priority-ceiling protocol using the priority transform interface. The ORB will have a record of object ID and the associated name for the object which is registered using the schedule-object method. The priority of the invocation up call is decided and set accordingly using inbound transform interface. Run time schedulers are implemented using ORB specific interceptors. Each node in the system has a shared memory segment, which is updated with application details such as names and associated priorities, priority ceiling etc. by the schedulability analyzer by invoking a factory object, which is pre-launched on each desired node.

6. Performance Experiments

The experiment evaluates the degree of priority inversion in the real-time MICO-MT ORB and non real-time MICO-MT ORB on QNX RTP by measuring the latency for highest priority task. The experiment was conducted with two machines, one P2 with 128 MB RAM which hosted the server and another P3 with 300 MB RAM running the client, both with QNX RTP

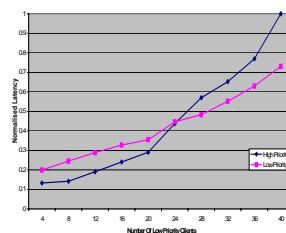


Figure 2a

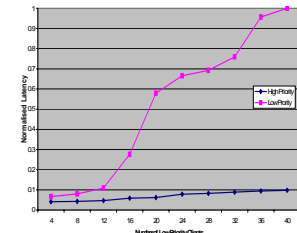


Figure 2b

Figure2a. Latency on MICO-MT ORB

Figure2b. Latency on RT MICO-MT ORB

connected through 100Mbps Ethernet. The test used clients with n different priorities. Each client sends invocations to the server periodically. The two way invocation latency of the highest and lowest priority

client is measured. All the requests are dispatched simultaneously using pthread_barrier. Figure 2 shows the results. It can be seen that, unlike the non real-time ORB (Figure 2a), in the real-time ORB (Figure 2b) irrespective of increase in the number of low priority clients the latency of highest priority client does not change significantly, which confirms that the ORB is deterministic with RT-CORBA extensions.

7. Industrial Application

Industrial units such as power plants use SCADA monitor and control power plant systems. Our SCADA system has many Remote Terminal Units (RTUs), which are distributed and gather system data as analog/digital inputs. Each RTU has a maximum of 14 I/O slots for analog/digital input/output signal conditioning boards. Each I/O board has 32 analog or digital inputs. The system demands temporal data consistency of 5msec and 50msec for digital and analog inputs respectively at RTU. The periodicity of the client requests varies between 1 second to 5 seconds. Process Computers, Man Machine Interfaces (MMI) servers, Alarm Processing module servers, Control Computers etc. constitute client nodes 1 to n.

Figure 3 depicts the SCADA architecture based on RT-CORBA. RTU software has two threads T2 and T1 with a periodicity of 5msec and 50msec which scans all digital and analog input boards respectively and updates the slot specific data of CORBA servants. The write operation on slot specific data is local critical section for these threads. The threads are triggered by POSIX timers. Each node (Node 1 to Node n) of SCADA can have one process (RT-CORBA client) per RTU and requests each slot data on a thread with a periodicity between 1 second and 5 seconds depending

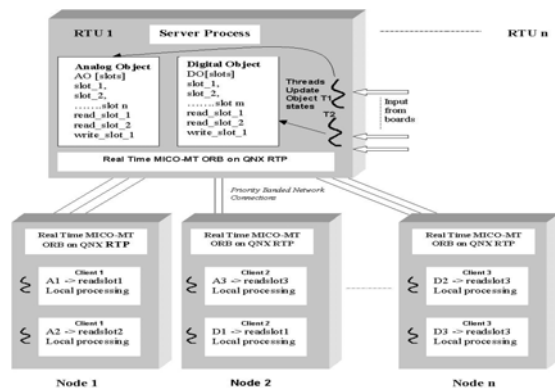


Figure3. RT-CORBA based SCADA

on the requirement. The read operation on slot specific data is global critical section for these threads. Each client thread may access data of multiple slots of an RTU also. System software is statically configured during build-time with schedulability analysis results.

The performance of SCADA was tested with two RTUs having 14 I/O boards on each RTU and three client nodes which are interconnected by Ethernet. The tasks of each client node were synchronized using pthread_barrier and released at the same instant. The sampling frequency of analog and digital inputs is 20Hz and 200Hz respectively. Few digital inputs were simulated by applying a pulse train with 15msec cycle time. Few analog inputs were simulated with slow varying signal with a frequency of 10Hz and decreased to 2Hz. The data collected at RTU was analyzed and found that digital input thread did not miss the pulse inputs.

8. Conclusion

An infrastructure consisting of enhanced MICO-MT ORB core with RT-CORBA APIs on QNX RTP, static scheduling service as part MICO-MT ORB core and schedulability analyzer for distributed and non-distributed systems on QNX RTP was developed. An embedded application was built using the infrastructure. Enhanced MICO-MT with real-time extension bounds the priority inversion, which is a prerequisite to employ CORBA middleware for real-time system applications. The performance fulfils the requirement of our SCADA.

Acknowledgements: We express our sincere gratitude to Mr. N. Mohan Ram, Chief Technology Officer, C-DAC, Bangalore for his encouragement in executing the R&D project on “Real-time Fault tolerant System for Industrial Application” and thank Industrial Application Division of Department of Information Technology, Government of India for granting R&D projects to Real-time Systems Group, C-DAC, Bangalore, India.

References

- [1] OMG, Real-time CORBA Joint Revised Submission, OMG Document ptc/99-05-03., May 1999
- [2] MICO-MT CORBA ORB: www.mico.org

- [3] Software Architecture for reducing priority inversion and non-determinism in real-time object request brokers, Douglas Schmidt et. al., Journal of Real-time System, 1999
- [4] Design and Performance of a Real-Time CORBA Scheduling Service, Christopher D. Gill, David L. Levine, and Douglas C. Schmidt, Distributed Object Computing, Washington University, August 1998
<http://www.cs.wustl.edu/~schmidt/TAO.html>
- [5] The Design and Performance of a Real-Time CORBA Event Service, Timothy H. Harrison, Carlos O’Ryan, David L. Levine, and Douglas C. Schmidt, Distributed Object Computing, Washington University
<http://www.cs.wustl.edu/~schmidt/TAO.html>
- [6] Synchronization in Real-Time Systems - A priority Inheritance Approach, Ragunathan Rajkumar, Kluwer Academic Publishers
- [7] Objective Interface Systems Inc.
<http://www.ois.com/products/prod-5.asp>
- [8] RapidSched: Static Scheduling and Analysis for Real-Time CORBA, L.Dipoppo, V.F Wolfe, University of RhodeIsland
<http://homepage.cs.uri.edu/research/rtisorac/publications.html>
- [9] A Practitioner’s Handbook for Real-Time Analysis, Mark Klein, Thomas Ralya et. al., Kluwer Academic Publishers.
- [10] Real-time Systems, Jane and Liu, Prentice Hall Inc.
- [11] Lehoczky, et. al. "The Rate Monotonic Scheduling Algorithm: Exact Characterization And Average Case Behavior", RealTime Systems Symposium, Dec. 1989.
- [12] Concurrent Control in Real-Time Object-Oriented Systems- The Affected Set Priority Ceiling Protocols by Squadrito, DiPippo, Cooper, Esibov and Wolfe. IEEE Symposium on Real-Time Object Oriented Computing, Kyoto, Japan, April 1998.