A Real-Time Total Order Multicast Protocol

Kayhan Erciyes[1] and Ahmet Şahan[2]

[1] California State University, San Marcos,
Computer Sci. Dept., 333 S.Twin Oaks Valley Rd.,
San Marcos CA 92096, U.S.A.
kerciyes@csusm.edu

[2] Ege University International Computer Inst.
35100 Izmir, Turkey
sahan@ube.ege.edu.tr

Abstract. We describe, analyze and submit results of a real-time total order multicast protocol developed on a distributed real-time system architecture that consists of hierarchical rings with synchronous packet delivering characteristics. The protocol is structured on and closely interacts with the distributed clock synchronization and the real-time group management modules. The synchronous characteristics of the protocol makes it suitable for hard real-time applications where total ordering is required. The complexity analysis of the protocol is given and the performance results are shown for several scenarios. We show that the developed protocol is correct, scalable and real-time ...

1 Introduction

A group is a logical name for a set of computing elements whose membership may change with time. Replication using process groups for fault tolerance has attracted many researchers for many years [3][4]. There are several systems which provide fault tolerant group communication such as Horus [9] and Totem [2]. Moshe [6] extends these services to a WAN. The common goal of these projects is to provide a reliable multicast communication for process groups. Most of these systems are event-driven or asynchronous systems and their suitability to a distributed real-time system such as a process control or a flight control system should be considered with care. For distributed real-time systems, synchronous systems which include periodic functionalities, can yield better performance and are usually preferred for communication in such systems [8].

Total Order Multicast (TOM) is the basic paradigm to provide message ordering in fault tolerant systems that use active replication [7]. TOM ensures that no pair of messages are delivered to the members of a group in a different order. The fundamental properties of TOM are *Validity*, *Uniform Agreement*, *Uniform Integrity* and *Uniform Total Order* [5]. *Atomic broadcast* is a special case of total order multicast where a TOM message is delivered to all of the group members or none. Atomic Broadcast or Reliable TOM protocols can be symmetric or asymmetric depending on whether some privileged nodes exist in the system or not. TOM has been studied extensively and many protocols have been proposed. A detailed survey is given in [5]. However, to our knowledge, there is not any significant research directed toward a real-time total ordering protocol.

The aim of this study is to revise and implement a distributed real-time system model that was designed previously [8] and then to design and implement a real-time total ordering protocol with distributed clock synchronization and group management modules. The model consists of hierarchical clusters of processing nodes which are connected by some communication medium. Tokens are delivered on hierarchical rings periodically enabling a synchronous communication. We also show that this model is suitable for any distributed real-time application that requires deadline guarantees and scalability. The paper is organized as follows. In Section 2, the base distributed real-time system model is described. The distributed clock synchronization and the total ordering protocol are described in sections 3 and 4. The analysis of the proposed architecture and the protocol are detailed in Section 5, the implementation results for regular cycle and total ordering protocol are given in Section 6 and the conclusions are outlined in the Conclusions section.

2 Distributed Real-Time System Architecture

The distributed real-time system model designed consists of hierarchical clusters of node members where each node represents a processor as shown in Fig. 1. Each cluster has a coordinator which is called the *Representative*. In the three layer design, two kinds of representatives provided are called as the *Sub-Representative* which is the coordinator for the lower ring and the central coordinator of rings that have Sub-Representatives members is the *Super-Representative*.

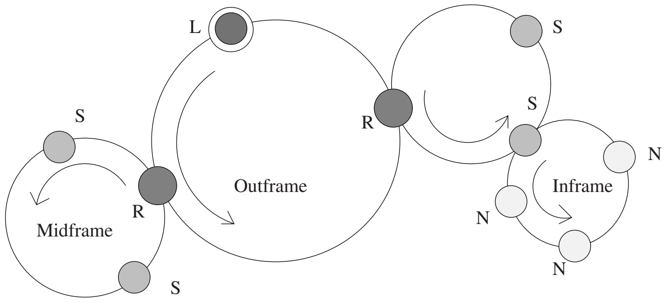

Fig. 1. The Distributed Real-time Model, L: Leader, R: Super-Representative, S: Sub-Representative, N: Node

At the highest level, the Leader is the coordinator and issues a token periodically called *outframe* on its ring, When Representatives receive the token, they also issue a token called *inframe* or *midframe* to collect data from their nodes.

Based on this protocol, the Total Order Multicast protocol is designed with the group management and the distributed clock synchronization modules as shown in Fig. 2. The Total Order protocol needs the clock synchronization service to sort the messages with respect to their valid timestamps and also the group management module to identify and manage atomicity of message delivery as detailed in sections 4 and 5.

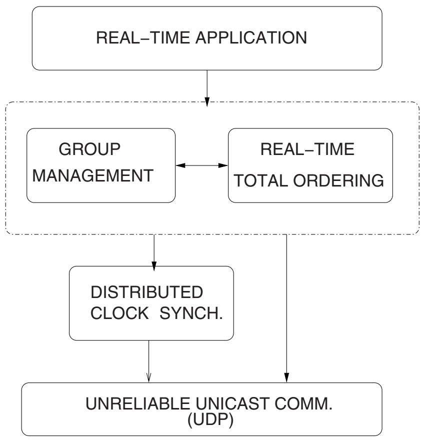

Fig. 2. The Real-time Total Order Architecture

3 Distributed Clock Synchronization

The clock synchronization is the base layer in many distributed systems where time services can be provided to any upper layer. The Total Order Multicast protocol proposed requires the clocks to be synchronized so that a global time frame is established. The three types of messages that can reach the nodes are COLLECT, SET or NONE. If NONE comes, it means increment your virtual clock value as frame period parameter and if COLLECT is received, the node writes its clock data into its slot in inframe. Finally, if SET is received, the node changes its clock value by adding a constant communication delay to the new coming value. When a COLLECT message comes from the Leader to a Representative, it collects values from the nodes, estimates an arithmetic average and sends it to the Leader. After the Leader issues a COLLECT message and receives all clock values, it also estimates an average value by adding the communication delay. The communication delay is related with frame issue and arrival time values. At the next period, it sends this new clock value to all by a new SET message. The finite state machine diagrams of the three processes for clock synchronization are depicted in Fig. 3.

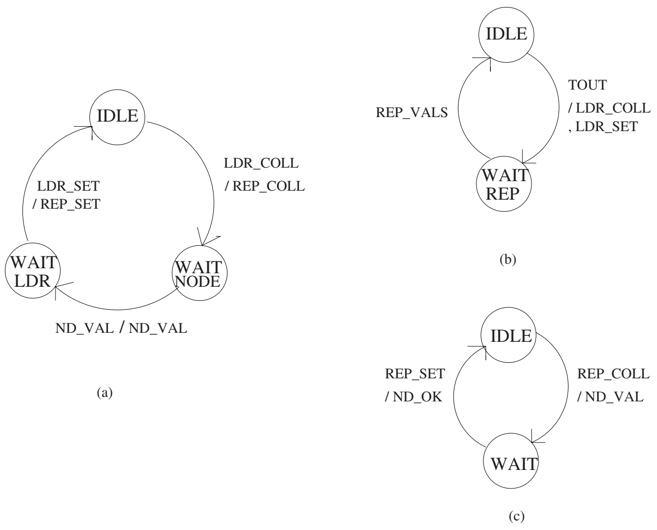

Fig. 3. FSM Diagrams of the Sub-Representative (a), the Leader (b) and the Node (c) for Clock Management

4 Total Order Multicast and Group Management

The Real-Time Total Order Multicast (RT-TOM) protocol is based on the normal operation of the protocol where the Leader functions as the central sequencer. It periodically gathers the requests from the Super-Representatives and sequences them with respect to their timestamps. The messages however are delivered to the nodes asynchronously as they are sent but only delivered to the application upon the received order sent by the Leader after sorting. The code for the Super or Sub-Representative is shown below.

The Total Order Representative

```
 process TO_Rep
 begin
    repeat
     msg=receive_msg();
     switch (msg.type):
        case NODE_MULTICAST : send_msg(node_msg, next_rep);
                                  send_msg(node_msg, first_node);
                                  insert_msg(node_msg, unsorted_list);
        case LDR_SOLICIT    : append(unsorted_list, ldr_msg);
                              send_msg(ldr_msg, next_rep);
        case LDR_ORDER      : send_msg(ldr_order, next_rep);
                              send_msg(ldr_order, first_node);
    until forever
end.
```

When a node sends a multicast message (NODE_MULTICAST), the representative broadcasts this message to its local ring and its upper ring. It also enqueues the identity of this message with its timestamp, to be sent to the Leader when the Leader request (LDR_SOLICIT) is received. The cycle is completed when the ordered identities of the messages are received from the leader (LDR_ORDER) in which case the order is broadcast in the local ring. The Total Order node process *TO_Node* delivers the message to the application with respect to the order sent by the central sequencer (Leader) only when the order is received. The actual delivery of the message to the *TO_Node* process however is performed independently as the message is circulated via the representatives without any interface from the Leader. Group Message Cycle is defined as atomic which provides that a group message is received by all or none of the members of the group. The atomicity control is managed by the Leader which checks the group member count with the coming acknowledgement message count from the nodes and if these do not match, the delivery is abandoned.

5 Analysis

The performance analysis of the message circulation for the *collect* operation of the three layer ring protocol should include the following

1. Distribution of the Leader message to the Super-Representatives : O_1
2. Distribution of Super-Representative message to Sub-Representatives : O_2
3. Distribution and the collection of the nodes information from the individual nodes at the Sub-Representatives: O_3
4. Collection of Sub-Representative information at Super-Representatives : O_4
5. Collection of the Super-Representatives information at the leader : O_5

Lemma 1. *Distribution and collecting of the leader message to/from the individual nodes (steps 1, 2, 3, 4 and 5 above) take $O_{ntime}(m)$ time where m is an upper bound on the number of nodes in the lowest cluster.*

Proof. Assume k, l and m are the upperbounds for the number of nodes in the outer, middle and inner rings respectively. The leader sends the outer ring message to the Super-Representatives in $O_1(k)$ steps as *all-to-all communication* in a unidirectional ring takes $O(n-1)$ where n is the number of nodes in the ring. Similarly, the Super-Representatives transfer this information to the Sub-Representatives in $O_2(l)$ steps in parallel with the other representatives. Finally, the Sub-Representatives will transfer and gather the information such as current clock values from the nodes in $O_3(m)$ steps. Gathering of this information at the Super and Sub-Representatives are similarly done in $O_4(m)$ and $O_5(l)$ steps. All of the five steps above take $T_{ntime}(2k+2l+m)$ time. For simplicity, let us consider that k, l and m are equal to each other. The total time complexity of the *collect* operation is then $O_{ntime}(m)$. It should also be noted that the minimum value for the frame period issued by the leader T_{lmin} should be approximately equal to $(k+l+m)t_m$ where t_m is the average message transfer time between a pair of consecutive nodes so that required data is available at the Super-Representative nodes when the next token is issued. The *collect* operation requires two frame periods $(2T_{lmin})$ and the *set* operation requires one frame period (T_{lmin}).

Lemma 2. *The total number of messages transferred during a normal ring operation is $O_{nmsg}(m^3)$ where m is an upper bound on the number of nodes in the inner ring.*

Proof. According to Lemma 1, the total number of messages transferred is $O_1(k)$ in the first step of data transfer. Each Super-Representative will then initiate transfer of l messages for a total of $O_2(kl)$ messages. In the third step, the total messages in transit are $O_3(klm)$. Similarly, the returning of the messages with valid information from the nodes require $O_4(kl)$ and $O_5(k)$ messages to reach the Leader. Assuming k, l and m are equal, the total complexity is then $O_{nmsg}(m^3)$.

Theorem 1. *The Speedup S obtained by the ring protocol with respect to a one level ring protocol is $O(m^2)$ where m is an upper bound in the number of nodes in the inner rings.*

Proof. If a single layer ring is constructed with the same number of nodes as a three layer ring, it would have klm or m^3 members. The total transfer time would then be $O_{otime}(m^3)$ for this one layer ring. The speedup obtained by the ring protocol is then :

$$S = O_{otime}(m^3)/O_{ntime}(m) = O(m^2) \tag{1}$$

Corollary 1. *Total time spent for message delivery by RT-TOM is $O_{rttom}(m)$ where m is an upperbound on the number of nodes in the inner ring.*

Proof. The total order multicast is performed by a *collect* operation ($2T_{lmin}$), followed by the *sort* operation and then the *set* operation (T_{lmin}) where the Leader sends the order to the individual nodes via the representatives. To perform atomicity a further step (T_{lmin}) to get and then a step to check acknowledgements followed by a *set* operation (T_{lmin}) is needed. Assuming sorting and acknowledgement checking are each done in T_{lmin} time in the worst case, the total time spent for RT-TOM is $T_{rttom}(7T_{lmin})$ and its complexity is $O_{rttom}(m)$.

6 Implementation and Results

The Clock Synchronization and RT-TOM in addition to the middle member representatives are implemented in the scope of this study, other parts are implemented in [1]. Nodes, Representatives and the Leader are implemented as POSIX threads on Sun OS v5.7 UNIX operating system running on DEC Alpha workstations where each thread can run on different workstations. The communication between member threads is performed by FIFO queues and UDP/IP sockets and the network environment was 100 Mbps FDDI backbone. The packet circulation test results during normal operation with the frame period T_l set to 170 ms are shown in Table. 1 for light load in the system. We can see that for the first case, the frame period is approximately equal to the sum of the individual

Table 1. Packet Circulation Tests (ms)

Total Ordinary Nodes	Configuration	Inframe	Midframe	Outframe
81	*Sup:3; Sub:3; Node:9*	114	35	27
54	*Sup:3;Sub:3;Node:6*	74	15	26

frame circulation values measured which means $k + l + m$ is the maximum value that can be obtained for this testbed, which is 15 in this case.

For the Total Order Multicast protocol, the total time for the delivery of ordered messages were measured with respect to the number of concurrent messages present in the whole network as shown in Fig. 4. Frame period was set to 170 ms, level count was three and the measurements were taken for 1, 5 and 10 concurrent messages in the system for varying total number of nodes. As seen from these values, message delivery times are similar, about 7*170 ms, as they depend only on the frame issue period. This would also mean that however the network is partitioned for a constant sum of k, l and m, whether as 3:3:9 or 5:5:5 nodes in the rings, we would still have the same message delivery time for RT-TOM. However, we would have 81 nodes in the first case, but 125 nodes in the second. The number of nodes, as long as their sum is smaller than the value required for that particular T_{lmin}, and the number of concurrent messages do not have any effect on the message delivery time for RT-TOM.

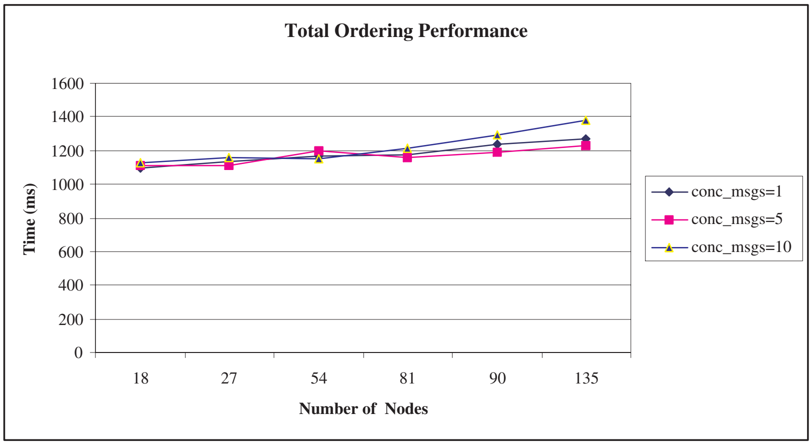

Fig. 4. Total Order Message Delivery Time

7 Conclusions

With this study, a synchronous, fault-tolerant, hierarchical and distributed real-time system model was designed and implemented. Using this model, a distributed clock synchronization module and a real-time total ordering protocol with the use of group management is realized. The main properties of the module are the total ordering of messages and atomicity where atomicity is performed by observing the group view. The main conclusions we draw are as follows. Due to its parallel operation in the rings, the normal ring operation and the RT-TOM protocol provides significant speedups with respect to a single level ring operation as stated in Theoerem 1. The performance for normal packet transfer obtained is scalable as shown by the test results. The RT-TOM protocol message delivery time is only dependent on the frame period, not on the number of concurrent messages in transit as stated by Corollary 1 and shown by the group message delivery tests. It is dependent on the number of nodes since token circulation period is dependent on the number of nodes, however, up to a predetermined value for the period under consideration, the message delivery time is almost constant as shown in Fig. 4. Further enhancements of RT-TOM are possible by piggybacking the acknowledgement from the nodes to the *set* token during atomicity checking phase to reduce message delivery time. The privileged nodes may fail and the recovery procedures are as in [8] which is not discussed here. Based on the foregoing, we can conclude that the protocol designed can be applied for TOM in hard real-time systems with stringent deadlines, possibly by decreasing the number of levels to 2 if the system under consideration is small.

References

1. Akay, O., Erciyes, K., A dynamic load balancing model for a distributed system , J. of Mathematical and Computational Applications ,8(1-3), 2003.
2. Y.Amir et all, The TOTEM single ring ordering and membership protocol, ACM Trans. Comp. Systems., 13(4),1995.
3. Birman K. P., van Renesse, R., Reliable Distributed Computing with the Isis Toolkit, IEEE Computer Society Press, Los Alamitos, Ca., 1994.
4. Chockler, G, Keidar, I., Vitenberg, R., Group communication specifications: a comprehensive study, ACM Computing Surveys, 33 (4), December 2001, pp. 427 - 469.
5. Defago, X., Agreement Related Problem: From semi-passive replication to Totally Ordered Broadcast. Ph.D. thesis, Ecole Polytech. Lausanne, Switzerland, 2000.
6. Keidar, I. et al, Moshe: A group membership service for WANs, ACM Transactions on Computer Systems (TOCS) 20 (3) , August 2002, 191-238.
7. Schenider, F., Replication management using the state-machine approach, *Distributed Systems*, pages 169-198, Ed. Sape Mullender, 2nd ed., 1993.
8. Tunali, T, Erciyes,K., Soysert, Z., A hierarchical fault-tolerant ring protocol for a distributed real-time system, Special issue of Parallel and Distributed Computing Practices on Parallel and Distributed Real-Time Systems, 2(1), 2000.
9. Van Renesse R., Birman K. P.,Maffeis S., Horus : A Flexible Group communication System, CACM Special section on Group Communication, 39(4), April 1996