Implementing the Multicast Inter-ORB Protocol

Alysson Neves Bessani[*], Joni da Silva Fraga[*], Lau Cheuk Lung[§]

[*] Laboratório de Controle e Microinformática (LCMI) – DAS – CTC – UFSC
Campus Universitário – Caixa Postal 476 – Trindade - CEP 88040-900 – Florianópolis – SC – Brazil
E-mail: {neves,fraga}@lcmi.ufsc.br

Graduate Program in Applied Computer Science
Exact Sciences and Technology Center
Pontifical Catholic University of Paraná - Curitiba - Paraná – Brazil
E-mail: lau@ppgia.pucpr.br

Abstract

This paper presents our experiments for integrating OMG MIOP (Multicast Inter-ORB Protocol) specifications into a CORBA ORB. We proposed an integration model which allows the coexistence of two different protocol stacks (IIOP/TCP/IP and MIOP/UDP/IP multicast), making possible a large spectrum of middleware support for distributed objects communication. That integration model is discussed in this paper, giving evidence of the compatibility of our approach with the CORBA specifications. We also do some considerations about the implementation of this model in a CORBA compliant ORB.

1. Introduction

The CORBA architecture (Common Object Request Broker Architecture) [1], introduced by OMG (Object Management Group), constitutes the most significant specification of *middleware* supporting distributed objects. Messages and formats in CORBA follow the General Inter ORB Protocol (GIOP) which is a generic protocol used for supporting remote method invocation. GIOP is actually a framework for a protocol which allows communications regardless of ORB implementations and consequences of a heterogeneous environment. The Internet Inter-ORB Protocol (IIOP) specifies how GIOP messages are exchanged using TCP/IP connections.

Although the IIOP and TCP/IP combination provides a reliable solution to point-to-point communications (handling error and flow controls, FIFO order, etc.) many other communication paradigms, when implemented over these protocols, may not be able to take full advantage of some important characteristics in the lower levels of a network. This difficulty is always reflected in the performance of these paradigms.

Many distributed applications depend on abstractions such as group communication for disseminating data among various receivers. Applications like distributed multimedia systems require a better utilization of network services. In 1999, the OMG published a RFP (*Request for Proposal*) in which defined a set of requirements for an unreliable multicast service that should be built on IP *multicast*. IP Multicast is an extension to the Internet Protocol (IP) that adds one to many communications to the IP services [2]. IP Multicast is characterized by the absence of guarantees and by the high performance, especially in local networks. In the middle of 2002, the OMG published the MIOP specifications (Multicast Inter-ORB Protocol) [3], which specifies the exchanging of GIOP messages on the stack UDP/*IP multicast*. MIOP is the key for making available an unreliable multicast service at CORBA.

The introduction of group communication in middleware standards has been target of various research projects and standardization proposals [2, 3, 4, 5]. For providing group communication to distributed applications, it is necessary a combination of protocols that deal with group management and group communication. Within the OMG, these facilities are separately standardized. Group management (failure detection, membership and state transfer) is treated in the FT-CORBA specification [6]. However, OMG has not yet published a specification, in the CORBA architecture, for group communication providing different levels of guarantees and semantics used in the usual group processing models. OMG is starting to deal with group communication. The first step was the creation of a task force for defining the specification of an *unreliable multicast* – the least restrictive model of group communication. Such initiative is expected to motivate new RFPs about group communication paradigms supplying more restrictive guarantees of agreement and order (e.g., reliable multicast, order FIFO, order causal, atomic multicast, etc).

The aim of this paper is to present our experiences in the integration of the MIOP into an ORB. Our integration model is discussed in this text, which shows evidence of the compliance of our approach to the CORBA specifications.

The text is organized in the following manner: in section 2 the paper presents a short description MIOP protocol and the corresponding object model. In section 3, we propose an integration model seeking to preserve two protocol stacks: one for point-to-point, and the other one for one-to-many communications. Details of implementation are described in section 4 and, finally, in section 5, the final considerations of this work are presented.

2. The MIOP Specification

The MIOP standard is divided in two main parts: the MIOP protocol and the MIOP object model to group comunication. The first part of this standard introduces the formats of MIOP packets used to encapsulate de GIOP messages; the second one defines the corresponding group model and abstractions for supporting this protocol, including the portable group reference used to address the UMIOP group members.

The MIOP protocol defines that a GIOP message is always fragmented and encapsulated in a set of MIOP packets. A MIOP packet is composed of a header – witch defines a set of fields for packet identification and message re-assembly at the receiver – and a fragment of the GIOP message. One of the requirements of the RFP which originated the MIOP specification is its integration with the UDP/IP multicast stack. Thus, the maximum size of a GIOP data fragment that can be contained in a MIOP packet usually depends on the size of the frame supported by the underling levels of the network.

Once a GIOP message is segmented into a set of MIOP packages (collection), they are multicasted to the group via UDP using IP services, in our case, IP multicast. The main features of IP mutlticast are: open group (it is not needed to be member to multicast a message into the group), unreliable communications (such as IP) and accessible via IP address class D (from `224.0.0.0` to `239.255.255.255`).

The MIOP unreliability, as well as UDP and IP multicast, is usually treated using timeout mechanisms for detecting the packet losses. When a MIOP packet is lost, the corresponding GIOP message is discarded.

The current CORBA objects model (stack IIOP/TCP/IP) specifies an Interoperable Object Reference (IOR) which is associated to a single implementation. The CORBA remote method invocation has support to exception handlers and implements the *at most once* semantic which adds reliability regarding the message delivery. Also, the FIFO ordering of messages is another characteristic of the CORBA remote method invocation.

The delivery of GIOP messages via an unreliable multicast service defines a communication context in a completely different way from that cited above. The objective in the UMIOP specification is to provide mechanisms for multicasting GIOP messages among distributed objects. In this context, an object group reference is introduced. Group reference is a different kind of IOR that contains information about how to access the corresponding group through the network. It is composed also of IP multicast address and port; but, due to the absence of object keys and host addresses for locating group members, the IP multicast address and port works like a logical identifier of the group. This logical identifier is used in the POAs (Portable Object Adapter[1]) which can then map the incoming MIOP messages, using this logical group identification, to the corresponding local members of the group.

3. The Integration Model

Figure 1 presents our integration model, implemented in the *MJaco* project. *MJaco* is an extension of *JacORB* [7], a conventional ORB/CORBA. The architecture *MJaco* is proposed to allow the co-existence of the two protocols stacks (IIOP/TCP/IP and MIOP/UDP/*IP Multicast*) in the same ORB, contributing in this way, for better interoperability and portability.

An ORB with two stacks of protocols is illustrated in figure 1: one for point-to-point communication, based on IIOP utilizing the TCP/IP services, and the other one is for multipoint communication formed by MIOP and UDP/IP *multicast*. Our integration model presents several elements introduced in the OMG specifications that composes the support which preserves the two communication paradigms. Other components and extensions were also added. Although they are not defined in the specifications, their purposes are to simplify the stacks integration and improve the efficiency of utilization of the set.

Our approach includes a local service object MGM+ that extends the MGM proposed in MIOP/OMG specification adding functionalities for group creation and management on the object level. Beyond creating group references – which are

[1] ORB interface is responsible for activating objects, and also for forwarding to them the received messages.

implemented through calls to ORB (assembling *corbaloc* URLs) – MGM+ registers created references on a name server, what makes the group registration automatically available to any other application[2]. MGM+ is also concerned with local changes of group members (management of local group membership); it is accomplished through some new operations, such as `add_member` or `delete_member`. These operations are implemented using POA group management methods.

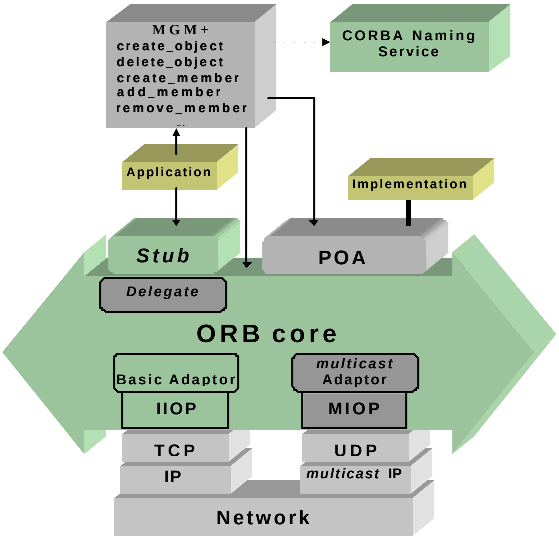

Figure 1. *MJaco* Architecture.

Another component that is part of our integration model is the Multicast Adapter, responsible for managing the *multicast sockets* utilized in the reception of MIOP packets and for delivering messages addressed to groups using the active POAs on the ORB. The MIOP module carries out the tasks, defined in the standard, which refers to the translation of the GIOP messages into collections of MIOP packets and vice-versa.

POA and *Delegate* are the main components of the ORB extended in our model. *Delegate* is modified in some points in order to support the sending of GIOP messages to groups; it is the first internal component of the ORB to be activated when a call to a method arrives on the *stub*. In our approach, it is decided at this point which stack of protocols will be utilized for sending a GIOP message corresponding to the method call activated. POA, in turn, besides the addition of the four primitives for group management described in the MIOP specification, must be modified for searching on Active Groups Map in order to obtain local group member implementations to whom the message is addressed. POA also activates Multicast Adapter (figure 1) for executing management operations of IP multicast. For example, when an `associate_reference_with_id`[3] operation is called for registering the group first member in ORB, POA activates Multicast Adapter to creates a socket and executes the IP multicast operation `JoinGroup` at the address defined on the UIPMC profile which is part of the group reference. After that, ORB is ready to receive messages addressed to this group. Other POA operations for supporting group communications are related with *IP Multicast* management operations.

4. *MJaco*: The Model Implementation

JacORB platform (http://www.jacorb.org) was selected to be used in our implementations of the proposed model (section 3t), based on the MIOP specifications. *JacORB* is an open source ORB, available by the *Freie Universitt Berlin*. This ORB was chosen based on its recognized quality and performance and also, due our own experience at using it in other two projects: *GroupPac* and *JaCoWeb* projects.

For processing MIOP messages, a new structure of classes was needed to be added at the main ORB architecture. Figure 2 presents an UML diagram with this structure.

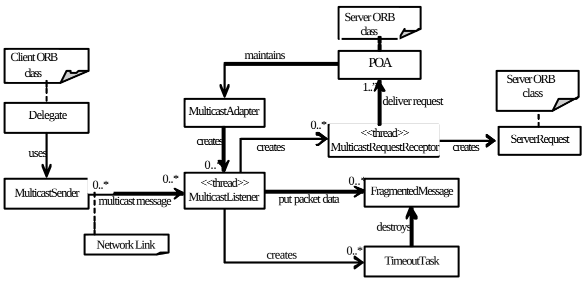

Figure 2. Classes of extensions for processing MIOP messages.

In this diagram, we identify *Delegate* class, into ORB, as a point of protocol stack selection. When the destination object corresponds is indicated by a group reference and the message is *oneway*, the processing is deviated to the class *MulticastSender*, which encapsulates the GIOP request into a collection of MIOP packets to be sent via IP Multicast. On the

[2] This must be previously specified in one of the object properties.

[3] One of the four new operations of POA that are defined in MIOP specifications. This operation associates local member implementations to the group reference (see example in figure 2).

server, the object of *MulticastListener* class is created for each group in which the ORB has members registered (one listener for each port and one port for each group) by the Multicast Adapter (*MulticastAdapter*). These listeners receive the MIOP packets and store the fragments of the GIOP message. When all the fragments are received, a thread (*MulticastRequestReceptor*) is activated for assembling the original message which is passed to all POAs of the ORB. Each POA, through the Active Groups Map (class *AGM*), translates the logical identification of the group – contained in the header of the GIOP message – with local addresses of members implementations of the group that were registered in the POA.

Note that the request is forwarded to all active POAs on the ORB, even those that do not register implementations of objects belonging to the destination group. This algorithm of message delivery was developed with the aim of making *AGM* the simplest possible and of decreasing, as much as possible, the overhead imposed by POA destination search.

5. Conclusions

The *Unreliable Multicast Inter-ORB Protocol* is the first step towards group communication in the CORBA. We are carrying out studies on reliable broadcast protocols and other group communication paradigms, and assessing ways they can be adapted to the CORBA, especially, using MIOP specifications, since the aim of the project is to implement over MIOP (on *MJaco*) group communication paradigms supplying more restrictive guarantees of agreement and order. For this reason, questions involving loss of packets, performance analysis, fault-tolerance, etc, were not discussed here. Our intention, in this paper, is to have a unreliable multicast support available into an ORB, where, together with the features of FT-CORBA group management, a set of functionalities would be formed to support different group models.

In this paper our solutions were presented for integrating UDP/IP multicast into a CORBA ORB. The integration model proposed does not jeopardize aspects of interoperability and portability of ORB as a whole. The ORB is capable of making invocations using both IIOP (conventional) and MIOP. This model can be easily adopted for integrating other protocol stacks, as long as they have an API available.

Furthermore, our experiences with *MJaco* implementations were also presented in this text. These developments were carried out over *JacORB*, a *Java free* ORB. *MJaco* was submitted to various tests. These tests point to some unfavorable costs to ORB in terms of performance. The results obtained show that there is a efficiency loss, about 60%, which may be compensated if we take into account the advantages of the distributed objects programming offered CORBA middlewares. The managing of objects and *threads*, completely through POA and ORB, along with the interoperability provided by the CORBA standard, add simplicity to applications that were formerly limited to the use of UDP *sockets* and other lower-level interfaces.

Our *MJaco* implementations, including the source code, are available on the Internet at the following address http://grouppac.sourceforge.net/ .

References

[1] Object Management Group, **The Common Object Request Broker Architecture Specification V2.6**, OMG Document. Available at www.omg.org, 2001.

[2] S. E. Deering, D. R. Cheriton, **Host Groups: A Multicast Extension to the Internet Protocol,**, Internet Engineering Task Force – RFC 966. Available in www.ietf.org 1985.

[3] Object Management Group, **Unreliable Multicast Inter-ORB Protocol Specification**, OMG Document. Available at www.omg.org, 2001.

[4] L. E. Moser, P. M. Melliar-Smith, P. Narasimhan, R. Koch and K. Berket, **A Multicast Group Communication Protocol, Engine, and Bridge for CORBA**, Concurrency and Computation Practice & Experience, vol. 13, no. 7, pp. 579-603, Jun 2001.

[5] P. Melliar-Smith, L. Moser and V. Agrawala, **Broadcast Protocols for Distributed System**, IEEE Transactions on Parallel and Distributed Systems, 1(1):17-25, Jan 1990.

[6] Object Management Group, **Fault-Tolerant CORBA Specification V1.0**, OMG document: ptc/2000-04-04, Available at www.omg.org. 2000.

[7] G. Brose, **JacORB: Implementation and Design of a Java ORB**, Procs. of DAIS'97, IFIP WG 6.1 International Working Conference on Distributed Applications and Interoperable Systems, Cottbus, Germany, Chapman & Hall, Sep. 1997.