

Implementation of CORBA/Java Environment in the Management of Mobile Communication Networks

Marek Malowidzki

Military Communication Institute

05-130 Zegrze, Poland

tel.: +48-22-688-55-96

fax: +48-22-774-38-65

e-mail: malowidz@cc.wil.waw.pl

Abstract - This paper describes the conception and implementation of CORBA/Java environment in mobile IP-based communication networks.

First the communication network, its architecture and specificity, is described briefly. Second, the reasons for using CORBA/Java are explained. Then the example of implementation is provided. Security issues are also reported. Finally, some conclusions about using CORBA/Java environment in mobile IP-based networks are made.

Key words: CORBA, Java, Network Management, Mobile IP Network

INTRODUCTION

Support for IP protocol in a mobile network enables the possibility of using commercial software, originally developed for the Internet, within such a network. Facilities and services offered by the software as well as technologies the software is based on may be used for many important purposes, e.g. for network planning and management, reporting, command control etc.

Mobile networks are especially difficult to manage as their structures are variable – nodes are still moving, radioline connections may change; generally speaking, the network is still in motion. This paper is devoted to the issue of distributed management system for mobile IP network.

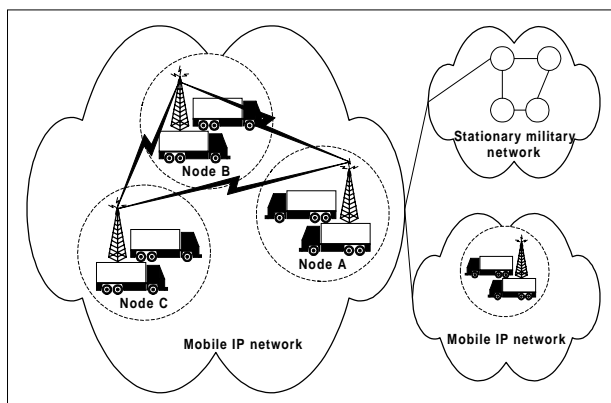


Fig. 1. Exemplary view of mobile IP network and cooperating networks

PROBLEM DESCRIPTION

Define distributed, object-oriented architecture for management system of a mobile IP network, with ability to interoperate with management systems of cooperating military networks. Detailed requirements should cover the following aspects:

Object-oriented architecture with strong interoperability capabilities: Management system should be based on an open, object-oriented architecture and be able to interoperate with management systems of cooperating networks.

Software portability: Software should be fully (both hardware and software) platform independent. This would allow using different hardware and software environments. This requirement could be omitted if a network uses uniform computer platform. However, in large and long-lasting projects it's worthy to take this requirement into consideration.

Security: Strong support for authorization, access control and secure communication between management system components is mandatory.

Flexibility of the structure: The architecture should support easy adding and removing components (for example, adding component for the management of a newly attached network node) and the possibility of management of the whole network from any place.

High survivability: The architecture should ensure good resistance to failures and damages and ability of the components to substitute the functionality of the damaged ones.

Information transparency, redundancy and distribution: Information should be distributed but easy to find and access. There ought to be no requirements for the clients to know the localization of components they want to communicate with. Information should be redundant to assure the resistance of damages and hostile attacks.

Relatively lightweight communication protocols: Protocols used within the management system should be able to operate in IP networks and generate low traffic.

PROPOSED SOLUTION

Three of the most popular object-oriented architectures: CORBA (Common Object Request Broker Architecture), DCOM (Distributed Common Object Model) and Java RMI (Java Remote Method Invocation) may be taken into consideration. They all allow invoking operations on remote objects and define portable communication protocols. However, RMI is strongly tied to Java environment as it uses many of Java-specific features. This limits interoperability capabilities – especially the possibility of integration with the other management systems, which can use different software environments. The other two architectures offer similar possibilities and there are fields where both have advantages and disadvantages. The comparison of CORBA and DCOM is beyond the scope of the paper and was well-covered in [3] (good technical comparison); there are also many articles published in the Internet. There's, however, at least one important issue where the advantage of using CORBA is obvious: presence of CORBA-based solutions in the market of network management. Many platforms and environments for network management support CORBA. CORBA and TMN (Telecommunications Management Network) integration is well specified (see [4]) and there are various tools available for bridging. So, the choice of CORBA seems to have an advantage in the field of interoperability with the management systems of cooperating networks.

Using CORBA architecture doesn't imply applying any particular programming language. There are mappings of IDL (Interface Definition Language, used for the definition of CORBA interfaces) defined for C, C++, Smalltalk, COBOL, Ada, Java and some educational projects aiming at implementing CORBA environment in other programming languages. The choice between C++ and Java, actually two of the most popular languages for CORBA programming, is not trivial. Java offers full portability, rich and simple in use API, powerful GUI (Swing) and good support for multithreading. However, C++ has proved its stability and suitability in many serious projects during last years while Java is relatively new technology, subject to changes in its basic API. Java implementations may suffer bugs (see, for example, [5] for description of security gaps in Java, corrected recently in JDK version 1.2.1). Besides, full portability forces separation from operating system, which affects efficiency. Despite of the drawbacks, the portability and simplicity offered by Java makes this language a promising choice. Besides, CORBA and Java technologies are going to integrate.

Let's have a closer view on how the choice of CORBA/Java technology meets foregoing requirements.

Object-oriented architecture with strong interoperability capabilities: CORBA is, of course, object-oriented technology. Management Information Model

defined in IDL language is programming language independent and existing mappings to programming languages assure free choice of programming environment used for the implementation of clients managing the network. This also ensures easy integration with management systems of cooperating telecommunication networks as many environments for network management support CORBA.

Software portability: As mentioned above, the choice of Java as a programming environment provides full software portability on almost all currently used platforms (for which Java Virtual Machine implementation is available).

Security: Most commercial implementations available on the market support Security Service with the use of IOP (Internet Inter-ORB Protocol) over SSL (Secure Socket Layer – see [6]). Security Service based on SSL protocol ensures:

- **Authorization:** made on the base of a chain of X.509 certificates and passwords known only to management staff. Cryptographic digest from user password serves as private key needed to decrypt the certificate and verify user identity.

- **Secure communication:** assured by using Secure Socket Layer. SSL uses symmetric cryptography (e.g. DES) for data communication, asymmetric cryptography (e.g. RSA) for authentication and key exchange and one-way hash function (digest - e.g. SHA) to ensure the integrity of messages.

- **Access control** must be supported by the implementation itself, although by separation of the clients (supporting various kinds of user interfaces) from servers (supporting means for management) this task is much simplified.

Relatively lightweight communication protocols: IOP over SSL seems to be a good solution for IP networks.

Flexibility of the structure, high survivability and information transparency, redundancy and distribution: must be supported by the implementation. However, as it will be shown in the following section, using CORBA can reduce the effort needed to meet these goals.

IMPLEMENTATION

The proposed implementation design is shown in Fig. 2. The management system consists of many CORBA servers, which cooperate with each other and exchange management information. The following part of this section discusses how the design meets the requirements mentioned above.

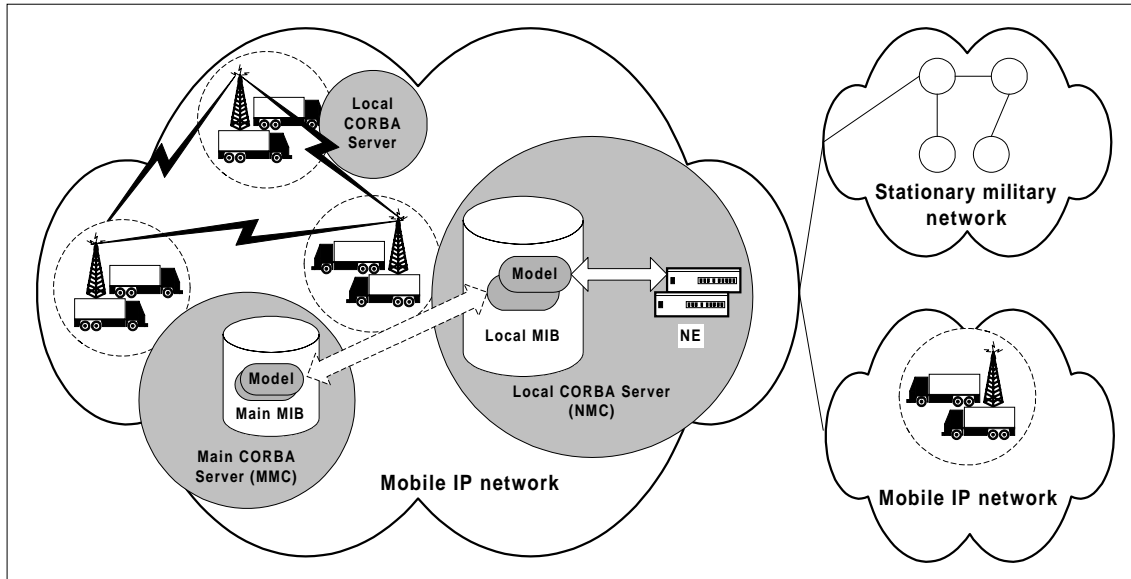


Fig. 2. Proposed implementation of the management system for mobile IP network

Main and local servers: To ensure the flexibility and survivability of the management system structure there must be many CORBA servers cooperating with each other across the network. Every network node is locally managed by its Local CORBA Server (LS) placed at the Node Management Center (NMC). LS contains all important information about the state of its node and holds and updates a Management Information Base (MIB) for the node. There are also Main Management Centers (MMCs) with Main CORBA Servers (MSs). MSs synchronize the local MIBs maintained by LSs and update their global MIBs containing information about the whole network. Thus the management information is not only distributed but redundant as well. MSs are ready to substitute those LSs, which don't work because of the damage or network failure.

LSs are basically designed for the management of local node resources. Using LSs allows local node management (without MS mediation) and reduces network traffic although the presence of LS is not required.

Servers cooperation: Each server monitors the state of cooperating servers. Every LS monitors MSs to check their availability and every MS checks the availability of every LS. The monitoring is performed by periodical polling. New servers (for example, running on newly attached nodes) are only required to know the localization of one main server. Servers exchange information about their actual presence.

Let's examine the case of a new node being attached. LS for this node notifies one of MSs about its presence in the network. The information about its presence is then sent to the other MSs. On the other hand, LS receives a list of working MSs. LS for newly attached node becomes an integral part of the management system.

Let's now have a look at more interesting situation of disabling one of working LSs. As soon as MS finds out about the lack of LS (during periodical polling), it decides to replace the functionality of disabled LS and becomes a server for that node.

The case when disabled LS (because of hardware failure or damage) is enabled is very similar to that case of new node attachment.

As explained above, the approach enables fast and – in most cases – even imperceptible redirection for the clients (see below) and trouble-free attachment of new nodes.

Memory Information Base and the management of network elements: Memory Information Base holds objects, which model real network resources such as communications devices, ports and trunks. MIB can be an object database, however it's often a relational database with some kind of mapping of rows of tables into objects. There's also one

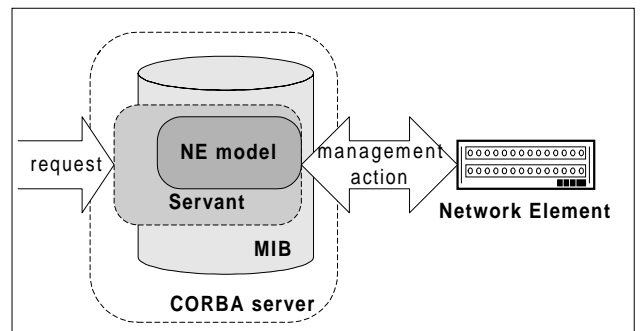


Fig. 3. Relations between an object in MIB, its CORBA servant and network resource

CORBA object (called servant) for every object in MIB. Performing request on a servant may result in action on managed entity (see Fig. 3).

However, MIB can contain thousands of objects, so it's impracticable or even impossible to create and keep servants for all objects in MIB for the server's lifetime. Another approach, using the facilities enabled by the Portable Object Adapter (POA), is suitable (see POA description in [1] and [7]). POA supports dynamic loading of servants on-demand and removing them when no more needed (for example, after some time they are unused). Thus, if POA detects a request to perform on a particular servant, it consults MIB to check the presence of corresponding object and starts the servant (if necessary) which the request is to be invoked on. To save the memory resources, servants may be removed just after performing the request or after some time they are unused for.

Dynamic loading and removing of servants are also enabled in some implementations of the Basic Object Adapter (BOA), the previous object adapter of CORBA. However, the concrete solution is implementation-dependent.

Client attachment and redirection: Clients can connect to a network and manage it from any place. The only thing required for managing the network is knowledge of the location of any working server and – of course – knowledge of private password needed for authorization. While performing operations on an object implementation, client will be redirected to the best server from communication point of view. Usually it will be the LS for the node where real managed entity is placed – LS contains the implementation object covering the managed entity. Fig. 4 depicts redirecting clients to the „nearest” server.

The redirection is supported by core CORBA facilities. When server A detects that a client should be redirected (1 – servant in server B should be used for performing a request), it just returns a special error (by throwing ForwardException - 2) which causes the client ORB to reissue the request to servant in server B (3 – see POA description in [1]) and results will be sent back by that servant (4). The process of redirection is imperceptible for the client.

SUMMARY

The problem of the definition of distributed, object-oriented architecture for management system of mobile IP network with ability to interoperate with management systems of cooperating military networks has been defined in the article. Solution based on CORBA/Java technology has been proposed, the reasons for this choice have been explained. Suitability of the proposed approach has been proved. Finally, implementation details have been presented and it has been shown that CORBA facilities simplify the effort needed to satisfy the requirements.

Despite of the fact that this paper doesn't cover all the issues related to network management (for example, event

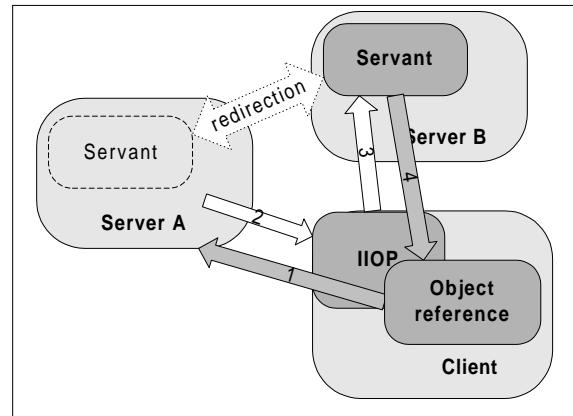


Fig. 4. Client redirection from servant in Server A to servant in Server B

reporting has been omitted), it discusses the fundamental problems with the design of distributed management system for mobile IP network and may be a base for further studies.

The practical implementation of management system based on the presented architecture is a goal of research project taking place in Military Communication Institute. The results will enable critical estimation of this proposition, especially at the aspect of feasibility of this approach in low-rate (dozens of kilobits) IP mobile networks with servers and clients running on PC machines.

REFERENCES

- [1] Object Management Group: *Common Object Request Broker Architecture and Specification*, version 2.2 <http://www.omg.org/library/c2indx.html>
- [2] Object Management Group: *CORBA services: Common Object Services Specification* <http://www.omg.org/library/csindx.html>
- [3] P.E.Chung, Y. Huang, S. Yaynik, D. Liang, J.C. Shih, C.Y. Wang, Y.M. Wing, *DCOM and CORBA: Side by Side, Step by Step, Layer by Layer*, C++ Report, V10N1, Jan 98 <http://akpublic.research.att.com/~ymwang/papers/HTML/DCOMnCORBA.html>
- [4] *JIDM Interaction Translation (Final Submission to OMG's CORBA/TMN Interworking RFP)* <http://www.omg.org/cgi-bin/doc?telecom/98-10-10.pdf>
- [5] Hostile Applets Home Page <http://www.rstcorp.com/hostile-applets>
- [6] *The SSL Protocol, Version 3.0* <http://home.netscape.com/eng/ssl3/ssl-toc.html>
- [7] D. C. Schmidt, S. Vinoski, *C++ Servant Managers for the Portable Object Adapter*, C++ Report, September 1998 <http://www.iona.com/hyplan/vinoski/col14.pdf>