

# Distributed Object Base Software

Hiroshi Yajima  
Hideki Ogawa  
Jun Nitta

*OVERVIEW: TPBroker is a distributed software system integrating new and legacy systems that, as Network Objectplaza's basic software, supports seamless linking between objects. Systems that incorporate this technology can make free use of objects no matter where they are located on the network. This means that a user in one department of a company can employ required functions and data from a system in another department without being aware of inherent differences with that other system, the physical location of data and programs, or the format or type of language that the data was created in. The Common Object Request Broker Architecture (CORBA<sup>\*</sup>) is an international standard that was developed in order to build such distributed systems. The ability to use TPBroker worldwide is assured by making it compliant with the CORBA specifications. Moreover, in an effort to promote more robust reliability and performance, we incorporated online transaction processing (OLTP) and other reliability and optimization techniques developed in the past as software elements, and thereby have achieved an open middleware solution that can also be applied to large-scale distributed systems.*

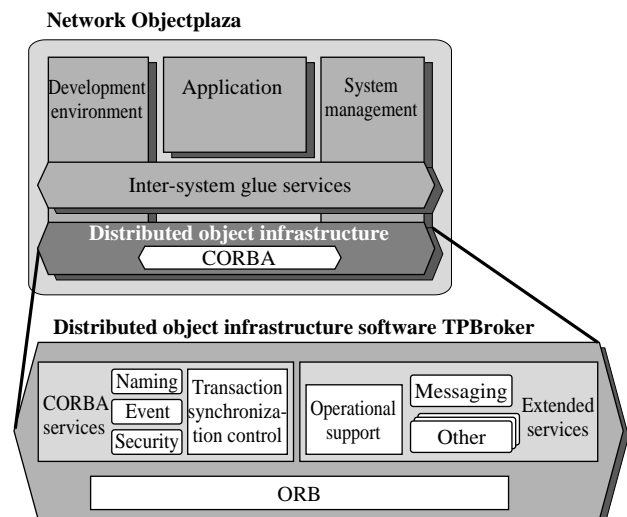
## INTRODUCTION

ALONG with the rapid expansion of the Internet, there is a growing trend observable at many companies to seek greater efficiency by integrating legacy systems, by going beyond the division and departmental level to share information on a company-wide basis, and by exchanging data with other companies. However, trying to implement such an object-oriented approach using conventional technologies is extremely costly in terms of system developer man-hours. This motivated us to develop the distributed object infrastructure software TPBroker in Network Objectplaza that can be used to support a diverse range of services providing a middleware solution to resolve these issues (refer to Fig. 1). By adopting this approach for the system platform, it essentially frees up system developers enabling them to devote their full attention to the development of critical business applications.

This article describes the main standardized technologies implementing the distributed object infrastructure software TPBroker, and outlines proprietary technologies permitting the application of TPBroker to large-scale basic business systems.

## CORBA STANDARDIZED TECHNOLOGIES

CORBA was one of the first specifications to be adopted by the Object Management Group (OMG<sup>TM</sup>), the standards body that was formed to develop and



ORB: object request broker

Fig. 1—Network Objectplaza and TPBroker.

In addition to being CORBA compliant, TPBroker is a software package that is functionally extensible and provides the high reliability and performance that is required of enterprise systems.

\* CORBA is a trademark or registered trademark of Object Management Group, Inc. in the U.S. and other countries.

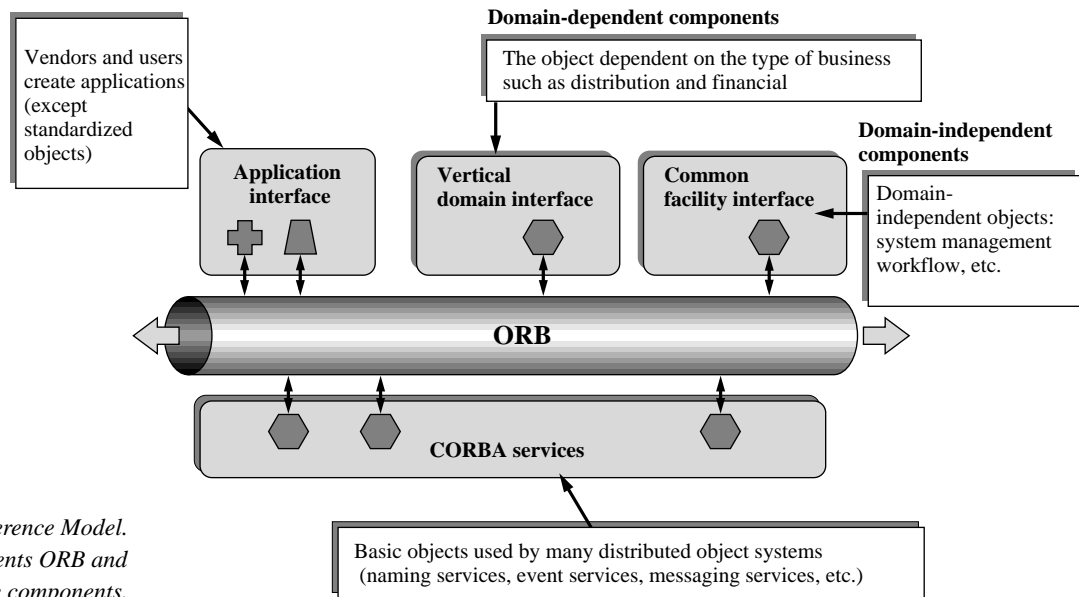


Fig. 2—CORBA Reference Model. TPBroker implements ORB and CORBA service components.

TABLE 1. TPBroker Agent Functions

There are three primary agents in TPBroker, all of which invoke request objects and execute objects independently.

Agent function	Basic processing	Purpose
Operational management	<ul style="list-style-type: none"> <li>Tracing, monitoring, and terminating objects</li> </ul>	<ul style="list-style-type: none"> <li>Improve operability</li> <li>Improve reliability</li> </ul>
Object management	<ul style="list-style-type: none"> <li>Object location management</li> <li>Automatic object connection, reconnection processing</li> </ul>	<ul style="list-style-type: none"> <li>Improve operability</li> <li>Improve performance</li> </ul>
Transactions	<ul style="list-style-type: none"> <li>Synchronization of distributed transaction termination processing</li> <li>Object restoration processing during abnormal</li> </ul>	<ul style="list-style-type: none"> <li>Improve performance</li> <li>Improve reliability</li> </ul>

promote standards relating to distributed object technologies. Object-oriented technologies run on distributed systems, and combine the advantages of both object-oriented programs and distributed computing.

Fig. 2 shows a schematic of the CORBA Reference Model. One will note from the Reference Model that all objects are linked by a mediator called the Object Request Broker (ORB™). Interfaces between objects are defined by the OMG Interface Definition Language that is specified by CORBA. By employing an ORB in conjunction with IDL, a diverse array of objects can be linked without any concern for where they are located on the network, differences in programming language, or disparate distributed system platforms.

### TPBROKER PROPRIETARY TECHNOLOGY

TPBroker is not only fully compliant with CORBA, it also achieves the high standards of reliability and performance required of a key system platform based upon technologies that are cultivated through actual system building. Let us take a closer look at two of these technologies: the agent function and asynchronous messaging function.

#### Agent Functions

TPBroker includes three main agents, as shown in Table 1. Each of these agents is capable of activating requesting objects and executing objects independently. The effects of the agents can be summarized in two points.

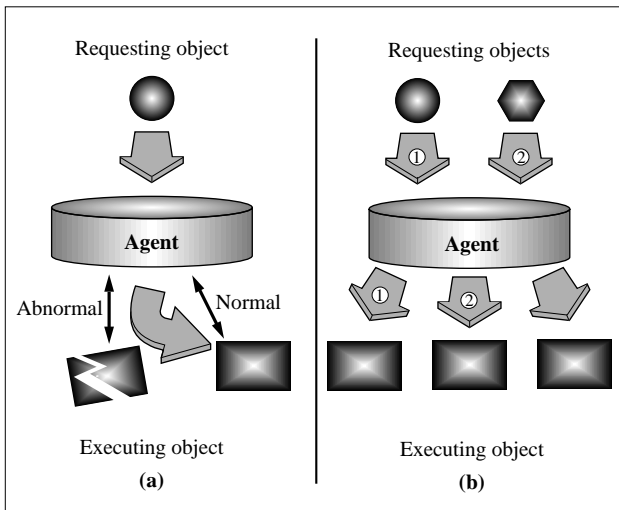


Fig. 3—Agent Tasks. Agents perform two essential tasks: (a) automatic switch-over to another executing object when a failure occurs, and (b) load distribution among executing objects.

(1) Enhanced reliability and operability

If an executing object fails, then one of two things happen: the agent either switches back to the requesting object and the executing object is restarted, or the requesting object is automatically reconnected to an equivalent executing object on a different server to lighten the load on the requesting object.

(2) Enhanced performance

Executing objects are monitored and requesting object requests are distributed among suitable executing objects to enhance processing performance. This is illustrated in Fig. 3.

Asynchronous Messaging Function

Distributed systems require two modes of communication: synchronous and asynchronous. In synchronous mode, transmission and receiving is in the form of speech with the destination party, so there is no question that the information reaches its destination. In this mode, however, information can only be transmitted and received when the far-side party is ready. For effective operation of a distributed system, it is necessary to implement a one-way mode of communication that does not involve synchronization with the destination party. In addition to the synchronous mode just described, CORBA also defines a one-way mode of communication that is similar to an asynchronous capability. However, this one-way mode is inadequate for systems demanding a high degree of reliability for two reasons:

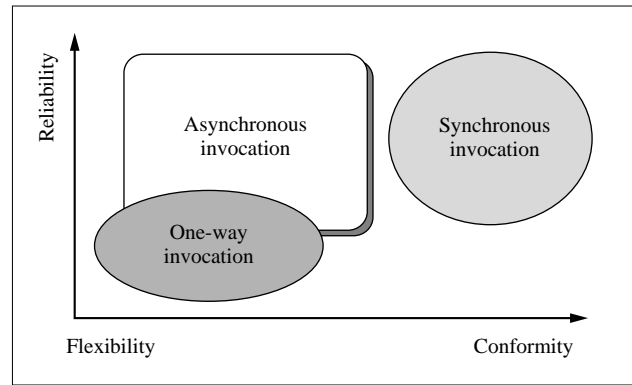


Fig. 4—Positioning of Message Invocation Modes. Asynchronous invocation is flexible and reliant; it plays an important role in large-scale distributed systems.

- (1) There is no way of knowing whether transmitted information is received at the other end.
- (2) There is no way of guaranteeing the delivery of information.

An asynchronous messaging function promises to overcome these restrictions. The relative positions of synchronous, one-way, and asynchronous modes in terms of reliability, flexibility, and high-speed performance is illustrated in Fig. 4.

TPBroker employs an Object Transaction Service (OTS) that is standardized in CORBA as the basis of its asynchronous messaging function, and the function has the following features:

Once a message is invoked, it is assured of being sent no matter what obstacles are encountered in route because it is recoverable.

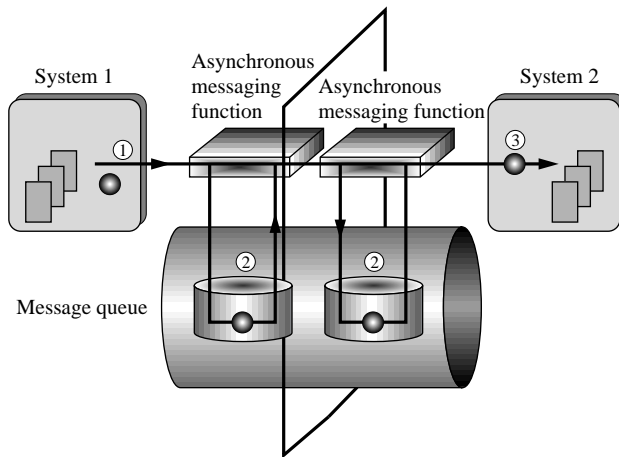
Invoked messages are guaranteed to reach their destinations and are not duplicated.

Programs that invoke messages have a means of verifying the transmission status of messages.

These capabilities are available not only between TPBroker sites, they can also be used with existing asynchronous messaging services such as OpenTPI message queuing. Fig. 5 shows an overview of the asynchronous message processing.

**FUTURE DEVELOPMENTS**

As use of the Internet and corporate intranets continues to grow, it is projected the distributed objects will be increasingly applied to data distribution and other key business systems, and will become increasingly important as a platform architecture for distributed systems. As a fundamental component of Hitachi’s Network Objectplaze system, cutting-edge



- ① Message sending  
Application programs can send messages regardless of destination party's status.
- ② Delivery of message over a network  
Messages are actually automatically stored in a database by the asynchronous messaging function. Then messages are sent between asynchronous messaging functions depending on the stage of the network.
- ③ Message receiving  
Messages are received when the destination-side application program is able to process the message.

Fig. 5—Asynchronous Messaging Function.  
Send and receive processing flow is handled according to procedures (1) through (3).

technologies will continue to be quickly incorporated into TPBroker in order to expand the range of actual running system applications. So, along with CORBA-compliant technologies, we will press ahead with innovative technologies required by key system projects, and recommend these advances for standardization in the OMG.

We are also focusing attention on another key objective of CORBA, interoperability with CORBA products developed by other companies. By improving the interoperability of CORBA products, this will also make it possible to implement more closely configured cooperative systems between companies that use CORBA-based products produced by other companies. In order to bolster the interoperability of TPBroker with other products further, we brought the system into compliance with the most recent version of CORBA 2.1, and are closely involved in interoperability demonstrations sponsored by the Distributed Object Promotion Group in Japan that was inaugurated in August 1997.

## CONCLUSIONS

This article has provided a broad overview of TPBroker that is based on the latest distributed object technologies. TPBroker has already proven its effectiveness through extensive use by numerous companies. We have submitted the key technologies to the OMG to be considered for standardization, while at the same time continuing to refine TPBroker even more. Revisions to the CORBA specifications are quickly integrated into the system. We will continue to pursue these various approaches in our efforts to make TPBroker a premier distributed object standard product.

## REFERENCES

- (1) OMG, "The Common Object Request Broker, Architecture and Specifications," OMG Doc. (1996).
- (2) OMG, "CORBA Services Specifications," OMG Doc. (1995).
- (3) T. Mowbray et al., Inside CORBA (1997).

## ABOUT THE AUTHORS



**Hiroshi Yajima**

Joined Hitachi, Ltd. in 1974 and now works at the Internet/Intranet Infrastructure Development Center of the Software Div. He is currently engaged in the development of distributed object platform products, and can be reached by e-mail at [yajima\\_h@soft.hitachi.co.jp](mailto:yajima_h@soft.hitachi.co.jp).



**Hideki Ogawa**

Joined Hitachi, Ltd. in 1984 and now works at the Internet/Intranet Infrastructure Development Center of the Software Div. He is currently engaged in the development of distributed object platform products, and can be reached by e-mail at [ogawa\\_hi@soft.hitachi.co.jp](mailto:ogawa_hi@soft.hitachi.co.jp).



**Jun Nitta**

Joined Hitachi, Ltd. in 1982 and now works at the Business & Information System Development Div. He is currently engaged in the development of distributed object platform products. Mr. Nitta is a member of the Information Processing Society of Japan, and can be reached by e-mail at [nitta@hi.com](mailto:nitta@hi.com).