INTERNATIONAL STANDARD

ISO/IEC 19500-2

First edition
2003-04-01

Information technology — Open Distributed Processing —

Part 2: General Inter-ORB Protocol (GIOP)/Internet Inter-ORB Protocol (IIOP)

Technologies de l'information — Traitement réparti ouvert —

Partie 2: «General Inter-ORB Protocol (GIOP)/Internet Inter-ORB Protocol (IIOP)»

Reference number
ISO/IEC 19500-2:2003(E)

PDF disclaimer

This PDF file may contain embedded typefaces. In accordance with Adobe's licensing policy, this file may be printed or viewed but shall not be edited unless the typefaces which are embedded are licensed to and installed on the computer performing the editing. In downloading this file, parties accept therein the responsibility of not infringing Adobe's licensing policy. The ISO Central Secretariat accepts no liability in this area.

Adobe is a trademark of Adobe Systems Incorporated.

Details of the software products used to create this PDF file can be found in the General Info relative to the file; the PDF-creation parameters were optimized for printing. Every care has been taken to ensure that the file is suitable for use by ISO member bodies. In the unlikely event that a problem relating to it is found, please inform the Central Secretariat at the address given below.

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org
Web www.iso.org

Published in Switzerland

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work. In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of the joint technical committee is to prepare International Standards. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

ISO/IEC 19500-2 was prepared by the Object Management Group (OMG) and was adopted, under the PAS procedure, by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, in parallel with its approval by national bodies of ISO and IEC.

ISO/IEC 19500 consists of the following parts, under the general title *Information technology — Open Distributed Processing*:

— *Part 2: General Inter-ORB Protocol (GIOP)/Internet Inter-ORB Protocol (IIOP)*

NOTE Other parts will be added in the future.

Introduction

The rapid growth of distributed processing has lead to a need for a coordinating framework for the standardization of Open Distributed Processing (ODP). ITU-T Recommendations X.901-904 | ISO/IEC 10746, the Reference Model of Open Distributed Processing (RM-ODP) provides such a framework. It defines an architecture within which support of distribution, interoperability and portability can be integrated.

Within the framework provided by the RM-ODP, ITU-T Rec. X.931 | ISO/IEC 14752, ODP - Protocol Support for Computational Interactions, defines how interactions between computational objects in a computational specification of a system relate to protocol support for those interactions in an engineering specification of that system.

Annex A to ITU-T Rec. X.931 | ISO/IEC 14752 defines a mapping to the General Inter-ORB Protocol (GIOP) and the Internet Inter-ORB Protocol (IIOP) which are specified by this International Standard.

GIOP is the base for all interoperability and support for all object request broker (ORB) functionality in the Common Object Request Broker Architecture (CORBA) specified by the Object Management Group (OMG). IIOP is the mapping of GIOP for the Internet.

Note: This document is technically aligned with the OMG CORBA GIOP and IIOP specifications.

Information technology — Open Distributed Processing —

Part 2: General Inter-ORB Protocol (GIOP)/Internet Inter-ORB Protocol (IIOP)

1 Scope

This standard specifies the General Inter-ORB Protocol (GIOP) for object request broker (ORB) interoperability. GIOP can be mapped onto any connection-oriented transport protocol that meets a minimal set of assumptions defined by this standard.

This standard also defines the Internet Inter-ORB Protocol (IIOP), a specific mapping of the GIOP which runs directly over connections that use the Internet Protocol and the Transmission Control Protocol (TCP/IP connections).

This standard provides a widely implemented and used particularization of ITU-T Rec. X.931 | ISO/IEC 14752, *Information technology — Open Distributed Processing — Protocol support for computational interactions*. It supports interoperability and location transparency in ODP systems.

2 Normative references

The following referenced documents are indispensable for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

2.1 Identical Recommendations | International Standards

- ITU-T Recommendation X.902 (1995) | ISO/IEC 10746-2:1996, *Information technology — Open Distributed Processing — Reference Model: Foundations*
- ITU-T Recommendation X.903 (1995) | ISO/IEC 10746-3:1996, *Information technology — Open Distributed Processing — Reference Model: Architecture*
- ITU-T Recommendation X.920 (1997) | ISO/IEC 14750:1999, *Information technology — Open Distributed Processing — Interface Definition Language*
- ISO/IEC 14752:2000, *Information technology — Open Distributed Processing — Protocol support for computational interactions*
- ISO/IEC 8859-1: 1998, *Information technology — 8-bit single-byte coded graphic character sets — Part 1: Latin alphabet No. 1*
- ISO/IEC 10646-1:2000, *Information technology — Universal Multiple-Octect Coded Character Set (UCS) — Part 1: Architecture and Basic Multilingual Plane*

2.2 Other Specifications

- CORBA 2.3 - The Common Object Request Broker: Architecture and Specification, Revision 2.3, Object Management Group, June 1999 (OMG Doc Number: Formal/98-12-01, ftp://ftp.omg.org/pub/docs/formal/98-12-01.pdf)
- CORBAservices: Common Object Services Specification, Object Management Group, December 1998 (OMG Doc Number: Formal/98-12-09, ftp://ftp.omg.org/pub/docs/formal/98-12-09.pdf)
- Java™ to IDL Language Mapping, Object Management Group, July 1999 (OMG Doc Number: Formal/99-07-59, ftp://ftp.omg.org/pub/docs/formal/99-07-59.pdf)
- STD 007 (also, RFC 793), Transmission Control Protocol, J. Postel, Internet Engineering Task Force, Sept. 1981
- STD 005 (also, RFC 791), Internet Protocol, J. Postel, Internet Engineering Task Force, Sept. 1981
- OSF Character and Code Set Registry, OSF DCE FRC 40.1 (Public Version), S. (Martin) O'Donnell, June 1994.
- RPC Runtime Support For I18N Characters — Functional Specification, OSF DCE SIG RFC 41.2, M. Romagna, R. Mackey, November 1994.

3 Definitions

For the purposes of this International Standard, the following definitions apply:

3.1 Recommendations | International Standards

This International Standard makes use of the following terms defined in ITU-T Rec. X.902 | ISO/IEC 10746-2:

behavior

interface

instance

object

state

type

This International Standard makes use of the following terms defined in ITU-T Rec. X.903 | ISO/IEC 10746-3:

operation

stub

3.2 Other Specifications

3.2.1 adapter

Same as object adapter.

3.2.2 Attribute

An identifiable association between an object and a value. An attribute **A** is made visable to clients as a pair of operations: **get_A** and **set_A**. Readonly attributes only generate a **get** operation.

3.2.3 client

The code or process that invokes an operation on an object.

3.2.4 data type

A categorization of values operation arguments, typically covering both behavior and representation (i.e., the traditional no-OO programming language notion of type.)

3.2.5 domain

A concept important to interoperability, it is a distinct scope, within which common characteristics are exhibited, common rules observed, and over which a distribution transparency is preserved.

3.2.6 dynamic invocation

Constructing and issuing a request whose signature is possibly not known until run-time.

3.2.7 dynamic skeleton

An interface-independent kind of skeleton, used by servers to handle requests whose signatures are possibly not known until run-time.

3.2.8 implementation

A definition that provides the information needed to create an object and allow the object to participate in providing an appropriate set of services. An implementation typically includes a description of the data structure used to represent the core state associated with an object, as well as definitions of the methods that access that data structure. It will also typically include information about the intended interface of the object.

3.2.9 interface repository

A storage place for interface information.

3.2.10 ORB core

The ORB component which moves a request from a client to the appropriate adapter for the target object.

3.2.11 repository

See interface repository and implementation repository.

3.2.12 request

A client issues a request to cause a service to be perfomed. A request consists of an operation and zero or more actual parameters.

3.2.13 results

The information returned to the client, which may include values as well as status information indicating that exceptional conditions were raised in attempting to perform the requested service.

3.2.14 server

A process implementing one or more operations on one or more objects.

3.2.15 signature

Defines the parameters of a given operation including their number order, data types, and passing mode; the results if any; and the possible outcomes (normal vs. exceptional) that might occur.

3.2.16 skeleton

The object-interface-specific ORB component which assists an object adapter in passing requests to particular methods.

3.2.17 synchronous request

A request where the client pauses to wiat for completion of the request. Contrast with deferred synchronous request and one-way request.

3.2.18 interface type

A type satisfied by any object that satisfies a particular interface.

3.2.19 interoperability

The ability for tow or more ORBs to cooperate to deliver requests to the proper object. Interoperating ORBs appear to a client to be a single ORB.

3.2.20 language binding or mapping

The means and conventions by which a programmer writing in a specific programming language accesses ORB capabilities.

3.2.21 method

An implementation of an operation. Code that may be executed to perform a requested service. Methods associated with an object may be structured into one or more programs.

3.2.22 object adapter

The ORB component which provides object reference, activation, and state related services to an object implementation. There may be different adapters provided for different kinds of implementations.

3.2.23 object implementation

Same as implementation.

3.2.24 object reference

A value that unambiguously identifies an object. Object references are never reused to identify another object.

3.2.25 objref

An abbreviation for object reference.

3.2.26 value

Any entity that may be a possible actual parameter in a request. Values that serve to identify objects are called object references.

3.3 Abbreviations

For the purposes of this International Standard, the following abbreviations apply:

ADT	Abstract Data Type
CCCS	Client Conversion Code Sets
CCS	Conversion Code Sets
CDR	Common Data Representation
CMIR	Client Makes it Right
CNCS	Client Native Code Set
CORBA	Common Object Request Broker Architecture
DCE	Distributed Computing Environment
OMG	Object Management Group
GIOP	General Inter-ORB Protocol
IDL	Interface Definition Language
IIOP	Internet Inter-ORB Protocol
IOR	Interoperable Object Reference
ORB	Object Request Broker
SCCS	Server Conversion Code Sets
SMIR	Server Makes It Right

SNCS	Server Native Code Set
TCS	Transmission Code Set
TCS-C	Char Transmission Code Set
TCS-W	Wchar Transmission Code Set
VSCID	Vender Service Context codeset ID

4 Introduction to GIOP/IIOP

This standard specifies the General Inter-ORB Protocol (GIOP) for object request broker (ORB) interoperability. GIOP can be mapped onto any connection-oriented transport protocol that meets a minimal set of assumptions. This standard also defines a specific mapping of the GIOP which runs directly over TCP/IP connections, called the Internet Inter-ORB Protocol (IIOP). The IIOP must be supported by conforming networked ORB products regardless of other aspects of its implementation. Such support does not require using it internally; conforming ORBs may also provide bridges to this protocol.

A definition of interoperability material necessary for the protocol is in clause 5; the definition of GIOP/IIOP is in clause 6. This standard is technically aligned with sections 13 and 15 of CORBA 2.3.

5 ORB Interoperability Architecture

For the purposes of this standard interoperability is defined as the ability for a client on ORB A to invode an IDL-defined operation on an object on ORB B, where ORB A and ORB B are independently developed. It further identifies general requirements including in particular:

- Ability for two vendors' ORBs to interoperate without prior knowledge of each other's implementations;
- Support of all ORB functionality;
- Preservation of content and semantics of ORB-specific information across ORB boundaries (for example, security).

In effect, the requirement is for invocations between client and server objects to be independent of whether they are on the same or different ORBs, and not to mandate fundamental modifications to existing ORB products.

5.1 Overview

5.1.1 Domains

The CORBA Object Model identifies various distribution transparencies that must be supported within a single ORB environment, such as location transparency. Elements of ORB functionality often correspond directly to such transparencies. Interoperability can be viewed as extending transparencies to span multiple ORBs.

In this architecture a *domain* is a distinct scope, within which certain common characteristics are exhibited and common rules are observed over which a distribution transparency is preserved. Thus, interoperability is fundamentally involved with transparently crossing such domain boundaries.

Domains tend to be either administrative or technological in nature, and need not correspond to the boundaries of an ORB installation. Administrative domains include naming domains, trust groups, resource management domains and other "run-time" characteristics of a system. Technology domains identify common protocols, syntaxes and similar "build-time" characteristics. In many cases, the need for technology domains derives from basic requirements of administrative domains.

Within a single ORB, most domains are likely to have similar scope to that of the ORB itself: common object references, network addresses, security mechanisms, and more. However, it is possible for there to be multiple domains of the same type supported by a given ORB: internal representation on different machine types, or security domains. Conversely, a domain may span several ORBs: similar network addresses may be used by different ORBs, type identifiers may be shared.

5.1.2 Bridging Domains

The abstract architecture describes ORB interoperability in terms of the translation required when an object request traverses domain boundaries. Conceptually, a mapping or *bridging mechanism* resides at the boundary between the domains, transforming requests expressed in terms of one domain's model into the model of the destination domain.

The concrete architecture identifies two approaches to inter-ORB bridging:

- At application level, allowing flexibility and portability.
- At ORB level, built into the ORB itself.

5.2 ORBs and ORB Services

The ORB Core is that part of the ORB which provides the basic representation of objects and the communication of requests. The ORB Core therefore supports the minimum functionality to enable a client to invoke an operation on a server object, with (some of) the distribution transparencies required by *CORBA*.

An object request may have implicit attributes which affect the way in which it is communicated - though not the way in which a client makes the request. These attributes include security, transactional capabilities, recovery, and replication. These features are provided by "ORB Services," which will in some ORBs be layered as internal services over the core, or in other cases be incorporated directly into an ORB's core. It is an aim of this specification to allow for new ORB Services to be defined in the future, without the need to modify or enhance this architecture.

Within a single ORB, ORB services required to communicate a request will be implemented and (implicitly) invoked in a private manner. For interoperability between ORBs, the ORB services used in the ORBs, and the correspondence between them, must be identified.

5.2.1 The Nature of ORB Services

ORB Services are invoked implicitly in the course of application-level interactions. ORB Services range from fundamental mechanisms such as reference resolution and message encoding to advanced features such as support for security, transactions, or replication.

An ORB Service is often related to a particular transparency. For example, message encoding – the marshaling and unmarshaling of the components of a request into and out of message buffers – provides transparency of the representation of the request. Similarly, reference resolution supports location transparency. Some transparencies, such as security, are supported by a combination of ORB Services and Object Services while others, such as replication, may involve interactions between ORB Services themselves.

ORB Services differ from Object Services in that they are positioned below the application and are invoked transparently to the application code. However, many ORB Services include components which correspond to conventional Object Services in that they are invoked explicitly by the application.