TECHNICAL REPORT

ISO/TR 24532

First edition
2006-06-01

Intelligent transport systems — Systems architecture, taxonomy and terminology — Using CORBA (Common Object Request Broker Architecture) in ITS standards, data registries and data dictionaries

Systèmes intelligents de transport — Architecture, taxinomie et terminologie des systèmes — Emploi de CORBA (Common Object Request Broker Architecture) dans les normes, registres de données et dictionnaires de données ITS

Reference number
ISO/TR 24532:2006(E)

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) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committees. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work. ISO collaborates closely with the International Electrotechnical Commission (IEC) on all matters of electrotechnical standardization.

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

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

In exceptional circumstances, when a technical committee has collected data of a different kind from that which is normally published as an International Standard ("state of the art", for example), it may decide by a simple majority vote of its participating members to publish a Technical Report. A Technical Report is entirely informative in nature and does not have to be reviewed until the data it provides are considered to be no longer valid or useful.

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

ISO/TR 24532 was prepared by Technical Committee ISO/TC 204, *Intelligent transport systems*.

Introduction

CORBA is one of many software technologies involved in distributed systems and system integration. There is a significant number of existing CORBA deployments in ITS, and discussions on best practice and standardization have naturally emerged, and discussion can often lead to comparisons between different technologies and confusion, even apparent “competition” between different software technologies.

The objective of this Technical Report is to identify the role of and provide guidelines for the use of CORBA in ITS.

Intelligent transport systems — Systems architecture, taxonomy and terminology — Using CORBA (Common Object Request Broker Architecture) in ITS standards, data registries and data dictionaries

1 Scope

This Technical Report clarifies the purpose of CORBA and its role in ITS. It provides some broad guidance on usage, and prepares the way for further ISO deliverables on the use of CORBA in ITS.

2 Terms and definitions

For the purposes of this document, the following terms and definitions apply.

2.1
General Inter ORB Protocol
inter ORB protocol that defines the message formats between ORBs in a distributed environment

2.2
Interface Definition Language
language for defining interfaces to CORBA objects which is independent of platform, operating system and programming language

2.3
Internet Inter ORB Protocol
inter ORB protocol that allows ORBs to use the Internet as a communications bus by mapping inter ORB messages onto TCP/IP

NOTE This is an implementation of GIOP.

2.4
Model-Driven Architecture
method of writing specifications and developing applications, based on a platform-independent model (PIM)

NOTE A complete MDA specification consists of a definitive platform-independent base UML model, plus one or more platform-specific models (PSM) and interface definition sets, each describing how the base model is implemented on a different middleware platform.

2.5
Object Request Broker
function within the CORBA architecture that acts as a broker in fulfilling client requests for services from objects in a distributed environment

2.6
Platform-Independent Model
model of a software system that is independent of the specific technological platform used to implement it

2.7
Platform-Specific Model
model of a software system that is linked specifically to a technological platform

2.8
Secure Sockets Layer
protocol for transmitting private information via the Internet by using public and private keys to encrypt data

2.9
Travel Information Highway
open and independent association of information publishers and receivers who have an interest in exchanging travel information using an agreed set of principles

3 Abbreviated terms

C2C	Centre to Centre
CORBA	Common Object Request Broker Architecture
GIOP	General Inter ORB Protocol
IDL	Interface Definition Language
IIOP	Internet Inter ORB Protocol
IOR	Interoperable Object Reference
ITS	Intelligent Transport Systems
MATTISSE	Midlands Advanced Transport Telematics Information Services and Strategies in Europe
MDA	Model-Driven Architecture
NTCIP	National Transportation Communications for ITS Protocol
OMG	Object Management Group
ORB	Object Request Broker
PIM	Platform-Independent Model
PSM	Platform-Specific Model
QMISS	Quantified Motorway Information Supply System
SSL	Secure Sockets Layer
TCP	Transmission Control Protocol
TCP/IP	Transmission Control Protocol/Internet Protocol
TIH	Travel Information Highway
UML	Unified Modelling Language

4 Requirements

4.1 CORBA Background

CORBA is a vendor-independent architecture and infrastructure that computer applications use to work together over networks. Using the standard protocol IIOP, a CORBA-based program from any vendor, on almost any computer, operating system, programming language or network can interoperate with a CORBA-based program from the same or another vendor, on almost any other computer, operating system, programming language and network.

CORBA applies object-oriented principles to distributed programming. A “CORBA object” offers services through well-defined interfaces specified using the OMG/ISO IDL. Clients use an object’s services by issuing requests to the object. The implementation details are kept hidden from clients. Language mappings from IDL to various programming languages make CORBA constructs available to invoke in programs.

This Technical Report does not attempt to provide a full explanation of CORBA. Key parts of CORBA are already International Standards, including ISO/IEC 14750 and ISO/IEC 19500-2.

CORBA is created and developed at the Object Management Group (OMG), a not-for-profit industry consortium. The best reference for further CORBA background is www.omg.org/corba.

4.2 When CORBA is appropriate

CORBA is a direct and productive way of implementing systems with distributed behaviour. Due to the wide range of language and operating system bindings available, CORBA is often a suitable choice when integrating existing systems. CORBA can provide a richer range of services than those available in many other middleware technologies.

4.3 Applying CORBA in ITS

For the purpose of this Technical Report, usage of CORBA will be split into distinct paradigms: “objects with behaviour” and “data/message transfer”. Both are legitimate usage paradigms for ITS.

4.3.1 Objects with behaviour

Distributed ITS systems have traditionally relied on messaging, but CORBA offers a richer programming model than just messaging. In this model, objects communicate and collaborate with one another to achieve the purpose of an overall system. Designers, rather than thinking about what data will be exchanged, consider what services will be offered by each component. The components are given CORBA interfaces with operations that denote some real behaviour.

Compared to messaging, this approach is more tightly coupled. Although asynchronous messaging is well supported in CORBA, the classic mode of use is for clients to make synchronous invocations of the operations of the service-providing CORBA objects. In many applications, this tight coupling is likely to be the best approach. Where the desired behaviour of components is known, creation of corresponding object interfaces is considered to be the most direct and productive way of designing and programming. The objects will tend to be domain-specific. A good example context in ITS would be integration of control systems, where components must interact to achieve overall system behaviour.

4.3.2 Data/message transfer[1)]

In a limited set of application contexts, data transfer is the best model. The reasons are partly non-technical, but nonetheless valid. Data owners, such as transportation authorities or operators, may have an idea that

1) While it is also possible to implement object invocations through messaging, this subclause describes message transfer in the sense widely used in ITS, where the messages are considered to be data.

their data is valuable, and that it could be used to provide further services, but they may not yet know exactly what services could be provided. Rather than waiting to define services, the data owner may actually achieve quicker uptake by making their data available, to encourage service providers to participate.

In travel information applications, both information service providers and network operators need to know the current state of the travel network. Because significant travel incidents are irregular and yet require timely response and information dissemination, there is often a requirement for asynchronous event-based data exchange. There is also a requirement for discovery of current state on initialization of a client application. These two distinct requirements are the key forces on the design of CORBA interfaces for travel information systems. While OMG has already standardized particular CORBA interfaces for common computing patterns ("CORBA Services"), none of the current OMG set provides a complete solution. For example, the OMG Notification Service has been used in ITS, but with an additional layer added to handle client initialization.

CORBA interfaces for data distribution will tend to be general, with operations phrased in terms of general software and data concepts rather than ITS-specific concepts. The ITS-specific content will be passed as parameters. An interesting design issue is whether specific ITS data models should be encoded into the IDL (e.g. as ITS-specific structs or value types) or whether the IDL should use general mechanisms (such as IDL "Any" type or IDL unions of possible basic types). Specific types have slightly better performance when marshalling. General types avoid recompilation of IDL after data model changes. While the great majority of applications would require re-coding anyway to reflect data model changes, general types are very useful to those few applications (typically graphical user interfaces or protocol bridges) that can adapt to new models at runtime using metadata.

Message transfer can be a useful technique where loose coupling is desired, perhaps to match underlying legacy system behaviour. In this case, existing OMG services such as the Notification Service can be employed.

4.3.3 Specialized CORBA versions

For embedded or other low-footprint systems, "Embedded CORBA" or "Minimum CORBA" shall be used. For hard real-time systems "Realtime CORBA" shall be used.

4.3.4 CORBA Security

Like any distributed technology, CORBA has points at which security threats should be considered. For each threat-point there are countermeasures, and many countermeasures have been standardized in OMG security specifications. Possibly the most widely implemented aspect of CORBA security is the use of the IIOP protocol over the secure SSL protocol. However, any deployment of CORBA (or indeed any information technology) should consider the threats, define a security policy, and (in the case of CORBA) investigate available countermeasures in CORBA security specifications and implementations.

4.3.5 CORBA Access Through Firewalls

CORBA has acquired a reputation for being awkward to use through firewalls. This is due to a combination of non-technical factors, which can be overcome.

A fundamental firewall function is to prevent any outgoing access except to specific ports with specific protocols, for example a firewall may explicitly allow the web protocol HTTP to pass through the well-known TCP port 80. CORBA IIOP also has established well-known ports (one for regular IIOP and one for IIOP over SSL). However, firewall administrators have been reluctant to open up ports other than the well-known HTTP ports. The argument is sometimes given that since HTTP is the protocol for retrieving static HTML information, while IIOP could be used to invoke any arbitrary behaviour, then the latter is less secure. However, this argument is refuted by the ability to "tunnel" — to layer virtually any protocol on top of an HTTP request and defeat the firewall. Even CORBA IIOP can be tunnelled using HTTP, and commercial products exist to implement this. This is clearly a ridiculous situation, since the act of layering IIOP on HTTP does not make it any more secure. It would be more secure to allow IIOP to pass but to impose further security restrictions. Unfortunately, firewall vendors have been slow to implement IIOP-aware firewalls that can interpret the details of IIOP requests and block any unauthorized invocations. However, there would have to be an existing security breach for an incoming unauthorized CORBA invocation to successfully find a target. The recommended strategy is therefore to allow IIOP access on the well-known port, as part of a carefully considered security implementation.

Firewalls also typically prevent connections being opened in the protected enclave. For this reason, CORBA supports a bi-directional mode of IIOP, in which callbacks are made on the same TCP/IP connection used for the initial outgoing communication. Bi-directional IIOP is currently supported by most ORBs, but if the feature is not supported, then the callback pattern can be avoided.

Firewalls also complicate addressing (true for all protocols) since the connections are made with firewalls before the ultimate endpoint. The recommended strategy for dealing with this (in the case of common TCP firewalls) is that the server ORB is configured to know about the inbound firewall, and uses the firewall address in the IOR that is supplied as an endpoint for clients to use. There is also a tag that can be added to IORs to declare inbound firewall details, but to take advantage of this feature, a GIOP-aware firewall is needed, and not all firewalls have implemented this feature.

It should be noted that firewalls are only one part of implementing an overall security policy, and CORBA security offers additional mechanisms.

4.4 Relationship of CORBA to other relevant technology

4.4.1 UML, CORBA and MDA

In all cases, it is good practice to start with a UML model. In the objects-with-behaviour paradigm, the objects and their interactions are modelled in UML, then the appropriate parts are realized as CORBA interfaces. However, when deciding on remote services, the designer needs to keep in mind performance issues of CORBA, and must choose a practical level of granularity. Theoretical "location transparency" does not mean that complexities and performance issues of distributed systems can be ignored at the design stage.

The use of a UML model that is independent of middleware implementation is one of the key principles of the MDA approach. A PIM contains no references to specific implementation technology such as CORBA, and is therefore usable as a basis for a range of implementations. In the MDA approach, a more detailed PSM is then created to specify the software that would be created using a particular middleware approach. For example, the same PIM could be mapped to both a CORBA PSM and an Enterprise Java Beans PSM. In many cases, the PSM is a UML model enriched with details so that it is a precise specification. In other cases, the PSM will consist of artefacts associated with the implementation technology. For example, with CORBA, IDL can be viewed as a form of PSM. However, IDL is less semantically rich than UML and the UML-based PSM is recommended. When creating CORBA PSMs, the OMG specification "UML profile for CORBA" should be used. If the pattern used in the software is a very standard one, then generators in MDA tools can automatically generate a PSM and code from a PIM.

For pure data distribution, the information model should be created in UML. This kind of model should be made precise, and stakeholders in systems should agree mappings from precise UML models to physical implementations. From such a precise UML information model, a developer will (with an understanding of the general delivery mechanism of the server and with reference to the agreed mapping) be able to develop an operational client application.

4.4.2 Links to OMG

OMG controls the evolution of the CORBA, UML and MDA specifications, and there is therefore a wealth of expertise on those technologies at OMG. OMG committees are divided into "Platform" (general software platforms) and "Domain" (industry-specific) groups. "Domain groups" include the "Transportation domain task force". Any CORBA-related specification is likely to benefit from the OMG adoption process, through exposure to OMG expertise.

For the data transfer paradigm, the precise UML models described above can be registered. The introduction of CORBA as the delivery mechanism makes no difference to the way that the underlying UML models are registered. A full discussion of the compatibility of UML with ISO14817 is beyond the scope of this Technical Report.

