An Introduction to OMG/CORBA

Wolfgang Emmerich

The City University
Dept. of Computer Science
London EC1V 0HB, UK
we@city.ac.uk

and

LogOn Technology Transfer
Frankfurter Str. 15
61476 Kronberg (Ts.), Germany
101575.1065@compuserve.com

ABSTRACT

An increasing number of applications are now being developed in a distributed setting. The main focus of this half-day tutorial is on OMG/CORBA, a widely recognised middleware standard for heterogeneous and distributed application integration. The tutorial discusses CORBA's object model and its representation in the OMG interface definition language (IDL). Programming language bindings to IDL are discussed; static and dynamic invocations are distinguished and CORBA services and facilities are sketched. The tutorial closes with an indication of recent standardisation efforts undertaken by the OMG.

Keywords:

Middleware, CORBA, IDL, Dynamic Invocation Interface, Interface Repository, Interoperability, CORBA Services, CORBA Facilities, Domain Interfaces.

PROBLEMS OF DISTRIBUTION

In a number of domains applications have to be adapted to evolving market conditions so quickly that it is no longer feasible to develop and, especially, to maintain huge, centralised applications. Applications, therefore, have to be down-sized into manageable components that can be developed and maintained autonomously and distributed over the network of an enterprise.

Distribution, however, imposes several challenges on application development. Autonomously implemented application components tend to be heterogeneous, being implemented in different programming languages, and are targeted for different hardware and operating system platforms. Moreover, if application components are distributed over a network, concurrency control issues must be solved and potential failures, such as temporarily unreachable components, must be addressed.

The need arises to support development of distributed applications using appropriate middleware layers, aiming at making the problems of distribution transparent to developers and users. Examples of such layers include the distributed computing environment (DCE) of the Open Software Foundation, ISO's ODP standard, various standards of the CCITT (such as X.400, X.500 and X.722), the Common Object Request Broker Architecture (CORBA) of the Object Management Group and the evolving Distributed Component Object Model (formerly called Network OLE) from Microsoft. In order to be able to design and implement distributed application architectures, software engineers will need to know the middleware components that they can rely on. This tutorial will provide an overview of the middleware standards and focus on OMG/CORBA.

CONTRIBUTION OF OMG/CORBA

There are about ten CORBA implementations commercially available. Although the tutorial briefly sketches the major products, the main focus is not on a particular product but rather on the standards adopted by the OMG and their rationales.

Object Request Broker: CORBA facilitates integration of distributed and heterogeneous applications and reuseability and portability of application components, on the basis of object technology. The central component of CORBA is an object request broker (ORB). It enables client objects to request operation executions from server objects. Client and server objects need not be implemented in the same programming language; they can be running on different types of operating systems and on different types of hardware platforms. The interoperability specification in revision 2.0 of the CORBA standard [1] even defines how operation requests are to be transferred to server objects connected to ORBs of different vendors.

Object Model: To accomplish integration in the context of such heterogeneity, the OMG has adopted a common object model [3]. It defines a set of object-oriented concepts and facilitates mappings of the CORBA object model to a large variety of programming languages. The concepts include attributes and operations, mul-

tiple inheritance, polymorphism, operation redefinition and exceptions.

Interface Definition Language: CORBA IDL is a programming language independent module interconnection language with constructs for all concepts of the common object model. Application builders use this IDL to define export interfaces of objects and to group these interfaces into more coarse-grained modules. The OMG has adopted standard language bindings to the IDL for C, C++, Smalltalk and Ada-95. Further bindings for Java and OO-Cobol are currently being considered for adoption.

Static vs. Dynamic Requests: Operation execution requests can be defined statically or dynamically. Client stubs for static requests, which are programming language dependent, are generated by an IDL compiler. Client objects request operation execution through stubs as local procedure calls in their respective programming languages. Asynchronous requests are implemented by invoking a stub in a new thread. Hence, operation requests are defined at compile time of the client object. The static definition of requests has the advantages that it is easy to use and that type-safeness and the availability of requested operations are checked at compile time by the respective compiler used for the client object. However, static invocations cannot be used if the interface of a server object is not yet defined at the time of the compiling the client. An example is an object browser that displays attribute values of objects. Such a browser should cope with objects whose interfaces are defined after the browser has been built. In these cases, the Dynamic Invocation Interface (DII) is used to compose operation requests at run-time and send requests to a server object.

Interface Repository: In order to be able to use the DII effectively, a repository is required to look up type information of objects at runtime. The object browser, for instance, has to perform a look up of attribute names and types before it can compose an invocation request for an operation to obtain an attribute value. The CORBA Interface Repository (IR) serves this purpose. Any CORBA object has an operation that returns a reference to its interface definition stored in the IR. It provides operations to obtain the attribute names and attribute types of an interface definition and operations and their signatures. Hence, the IR is typically used to accomplish safe usage of the DII.

CORBAservices: A number of fundamental requirements common to all heterogeneous and distributed computing architectures are addressed by CORBAservices [2]. Fifteen services have been adopted so far and these include the definition of external object names (Naming), distribution of events among multiple objects (Event Notification), maintaining relationships between objects (Relationship), creation, migration and removal of objects (Lifecycle), persistent storage of objects (Persistence), atomic transactions (Transaction), and object concurrency control (Concurrency).

CORBAfacilities: The CORBAfacilities are layered on top of the services. Facilities are components that are useful horizontally across domain borders. The first facility adopted by the OMG is the distributed document component facility that evolved from the OpenDoc Framework produced by Apple, IBM, CI Labs and Novell. It features document components, their composition into compound documents and their portable visualisation across various user interface platforms.

FUTURE DEVELOPMENT OF CORBA

The OMG is now working on standards for different vertical market domains. The domains currently being considered are business objects, electronic commerce, health care, manufacturing, telecommunication and the financial sector. Domain specific task forces have been set up and are working towards domain interfaces. The financial domain task force, for instance, is currently working towards the specification of facilities for money and currency, international business calendars and international funds transfer. The tutorial will provide a brief overview of these areas so that attendees will be able to get a better understanding of future OMG technology adoptions.

Among desktop environments, Microsoft's OLE (object linking and embedding) has a major role. It is desirable to have an interoperability bridge between OLE and CORBA to enable Microsoft Windows clients to act as front-ends in a heterogeneous and distributed environment. The OMG has recently adopted the first part of the COM/CORBA interworking specification. This defines the mapping of Microsoft's common object model (COM) to CORBA's object model. The second part of this interworking specification will then define interoperability between OLE and CORBA, which will enable OLE objects to request operation execution from CORBA objects and vice versa.

REFERENCES

[1] *The Common Object Request Broker: Architecture and Specification Revision 2.0.* 492 Old Connecticut Path, Framingham, MA 01701, USA, July 1995.

[2] *CORBAservices: Common Object Services Specification, Revised Edition.* 492 Old Connecticut Path, Framingham, MA 01701, USA, March 1996.

[3] R. M Soley, editor. Object Management Architecture Guide. Technical report, Object Management Group, 492 Old Connecticut Path, Framingham, MA 01701, USA, 1992.