GOA COLLEGE OF ENGINEERING

"Bhausaheb Bandodkar Technical Education Complex"

Experiment No: 6

Case Study On Corba

Aim: Perform case study on Corba.

Theory:

History

- The CORBA (Common Object Request Broker Architecture) specification was developed in 1991 by the Object Management Group (OMG).
- The OMG was founded by eleven corporations to develop CORBA.
- CORBA 2.0 specification (introduced in 1996) permits communications between implementations made by different developers.

Goal

- The OMG's goal was to adopt distributed object systems that utilize object-oriented programming for distributed systems.
- Systems to be built on heterogeneous hardware, networks, operating systems and programming languages.
- The distributed objects would be implemented in various programming languages and still be able to communicate with each other.

Definitions

- CDR - Common Data Representation: An external data representation of all data types that can be used as arguments or return values in remote invocations.
- CORBA - Common Object Request Broker Architecture: An Object Request Broker (ORB) architecture specification for distributed object systems.
- GIOP - General Inter-ORB Protocol: CORBA 2.0 standard that enables different developer implementations to communicate with each other.
- IDL - Interface Definition Language: An interfacing programming language that allows communication across a heterogeneous distributed system.
- IIOP - Internet Inter-ORB Protocol: Internet version of GIOP that uses the TCP/IP protocol.
- Marshaling: The process of taking a collection of data items and assembling them into a form suitable for transmission in a message.
- Middleware: A software layer that provides a programming abstraction as well as masking the heterogeneity of the underlying networks, hardware, operating systems, and programming languages.
- OMG - Object Management Group: The organization that developed and maintains CORBA

- POS - Persistent Object Service: A persistent object store for CORBA where objects that live between processes activations can be stored.

Features

- CORBA consists of a language independent RMI
- Consists of a set of generic services useful for distributed applications.
- The CORBA RMI acts as a "universal translator" that permits client processes to invoke a method or process that may reside on a different operating system or hardware, or implemented via a different programming language.

CORBA RMI Features

- The CORBA RMI consists of the following main components:
- An interface definition language (IDL)
- An architecture (discussed in **Structure**)
- The General Inter-ORB Protocol (GIOP)
- The Internet Inter-ORB Protocol (IIOP)

CORBA IDL Features

- Provides an interface consisting of a name and a set of methods that a client can request.
- IDL supports fifteen primitive types, constructed types and a special type called *Object.*
- Primitive types: *short, long, unsigned short, unsigned long, float, double, char, Boolean, octet*, and *any*.
- Constructed types such as arrays and sequences must be defined using *typedefs* and passed by value.
- Interfaces and other IDL type definitions can be grouped into logical units called modules.

GIOP and IIOP Features

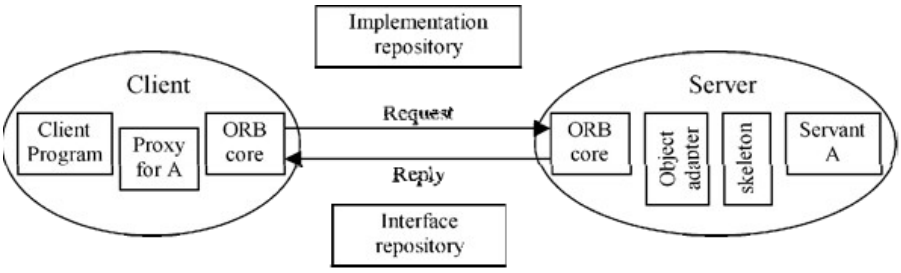
- GIOP: General Inter-ORB Protocol are the standards (included in CORBA 2.0), which enable implementations to communicate with each other regardless of who developed it.
- IIOP: Internet Inter-ORB Protocol is an implementation of GIOP that uses the TCP/IP protocol for the Internet.

CORBA Services

- Set of generic service specifications useful for distributed applications.
- CORBA Naming Service - essential to any ORB
- CORBA Event Service - define interfaces

- CORBA Notification Service - extension of event service
- CORBA Security Service - controls access
- CORBA Trading Service - allows location by attribute
- CORBA Transaction and Concurrency Control Service
- CORBA Persistent Object Service

CORBA RMI Structure



Structure - ORB Core

- ORB core
- Carries out the request-reply protocol between client and server.
- Provide operations that enable processes to be started and stopped.
- Provide operations to convert between remote object references and strings.

Structure - Object Adapter

- Object Adapter (server)
- Bridges the gap between CORBA objects and the programming language interfaces of the servant classes.
- Creates remoter object references for the CORBA objects
- Dispatches each RMI to the appropriate servant class via a skeleton, and activates objects.
- Assigns a unique name to itself and each object
- Called the Portable Object Adapter in CORBA 2.0
- **processes can run on ORB's produced by different developers.**

Structure - Skeletons and Proxies

- Skeletons (server)
- An IDL compiler generates skeleton classes in the server's language.
- Dispatch RMI's to the appropriate servant class.
- Client Proxies / Stubs
- Generated by an IDL compiler in the client language.

- A proxy class is created for object-oriented languages
- Stub procedures are created for procedural languages.
- Both are responsible for marshalling and unmarshalling arguments, results and exceptions.

Structure - Repositories

- Implementation Repository
- Activates registered servers on demand and locates servers that are currently running.
- Interface Repository.
- Provides information about registered IDL interfaces to the clients and servers that require it. Optional for static invocation; required for dynamic invocation.

How to use CORBA

- Server must include IDL interfaces in the form of servant classes.
- An interface compiler generates:
- the program (Java or C++) interfaces
- server skeletons for each IDL interface
- proxy classes (or client stubs) for each IDL interface
- A Java / C++ class for IDL defined *struct*
- helper classes for each IDL defined type
- Server
- Creates and initializes the ORB
- Creates an instance of servant class, which is registered with the ORB. Servant class extends the corresponding skeleton class and implementation methods of an IDL interface.
- Makes a CORBA object
- Client
- Creates and initializes the ORB
- Contacts Naming service to get reference to the server
- Invokes methods on the server

Applications of CORBA

- Used primarily as a remote method invocation of a distributed client - server system.
- Can communicate between clients and servers on different operating systems and implemented by different programming languages (Java cannot do this).
- Has many standards and services useful in implementing distributed applications.

- Process can be both server and client to another server
- Ideal for a heterogeneous distributed system like the Internet.

Significant Points

- Harder to use than a single language RMI such as JAVA.
- Compatible with various languages and operating systems.
- CORBA's IDL is the crucial part because it permits communication on a heterogeneous distributed system.
- The OMG consists of several companies that maintain compatibility and standardization of the CORBA system.

Summary

- The OMG developed CORBA for distributed object systems that will work on a heterogeneous distributed system.
- CORBA consists of many parts and services that are useful for distributed applications.
- The IDL portion is the crucial part that makes CORBA a language independent middleware.
- CORBA includes generic service specifications useful in implementing distributed applications.

Conclusion: Case study on Corba was successfully done.