CORBA in Component-Based Development

1. Introduction

CORBA, which stands for Common Object Request Broker Architecture, is a component-based development technology that enables interoperability among distributed objects, it is a standard developed by the Object Management Group (OMG) and is considered the world's leading middleware solution for the exchange of information across diverse hardware platforms, programming languages, and operating systems

The goal of OMG is to create a market for component-based software by accelerating the introduction of standardized object software. CORBA relies on Object technology; it is the world's first middleware solution that allows the exchange of information, regardless of hardware platforms, programming languages, and operating systems. Three important characteristics are exploited by clients using CORBA, which are:

- Invocation transparency.
- Implementation transparency.
- Location transparency.

2. The architecture of CORBA

The architecture of CORBA (Common Object Request Broker Architecture) is crucial for understanding its functionality and interoperability. Here's a breakdown of the key components

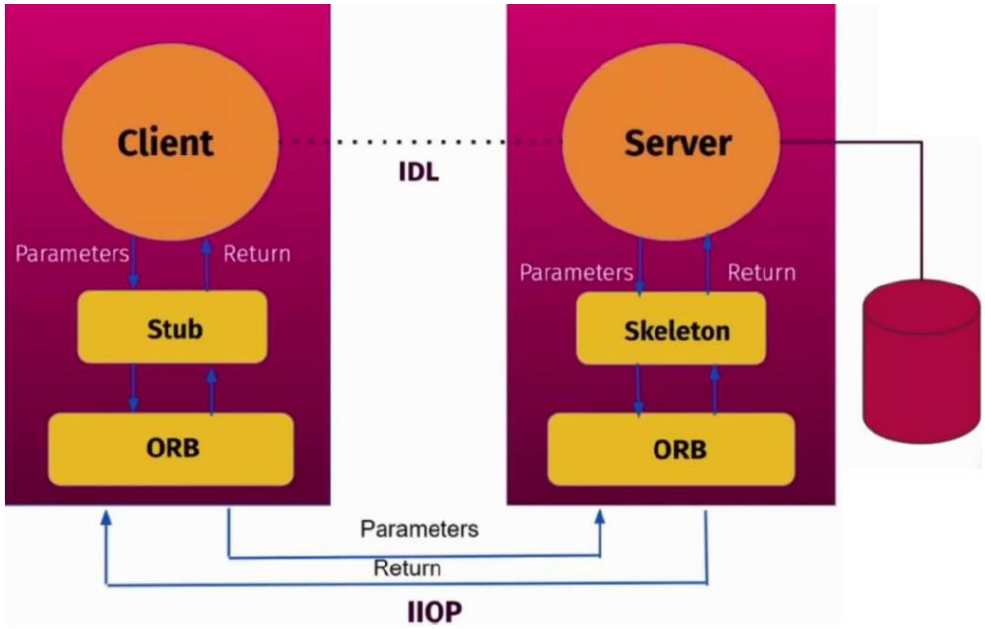

Fig.1: the architecture of CORBA

- **Interface Definition Language** (IDL), very similar to C++, which allows programmers to write an interface in a common language for all distributed objects. IDL generates a stub for the client and a skeleton for the server.
- **IIOP** or Internet Inter-ORB Protocol is a communication protocol created especially for ORBs, making it the standard protocol for communication between objects in CORBA.
- **ORB**: The heart of the CORBA architecture is the ORB (Object Request Broker). Each machine involved in CORBA must have an ORB running so that the processes on that machine can interact with the CORBA objects running in remote processes.

  The client ORB provides a stub for a remote object. Requests made on the stub are transferred from the client's ORB to the ORB serving the target object's implementation.
- **SKELETON:** The skeleton is the object-specific interface that is remotely exported via CORBA
- **IIOP**: The CORBA standard includes specifications for inter-ORB communication protocols that transmit object requests between various ORBs running on the network. The protocols are independent of the specific ORB implementations running on each side.

  The inter-ORB protocol delivers messages between two ORBs; these messages can be method requests, return values, error messages, etc. For two ORBs to communicate with each other, it is necessary to ensure that they both use the same protocol (IIOP).

  The IIOP protocol is an inter-ORB protocol based on TCP/IP and is therefore widely used on the Internet.
- **Services in CORBA:** CORBA services are object management services; they define basic services and a support structure useful for a wide range of applications. There are several services in CORBA, including: Naming Service, Event Service and Notification Service, Security service, Transaction service, concurrency control service, Trading service, Persistent state service, Life cycle service, etc. The most used services, considered fundamental, are:
  - **Naming Service**: the CORBA naming service provides a directory naming structure for remote objects. The CORBA naming service is one of the naming and directory services supported by JNDI (Java Naming and Directory Interface).
  - **Security service:** the security service is one of the most complex CORBA services. It consists of two levels:
    - Level 1 provides basic security for user authentication and invocation security, allowing applications to ignore the security requirements of the system; the ORB will handle operations in complete ignorance of the objects involved in an invocation.

- Level 2 offers everything that level 1 allows but with many more requirements: It allows a system to fully utilize all features (thanks to the use of APIs defined by the different interfaces of the security service).

- **Persistent state service**: it stores and retrieves objects. The persistent state service relies on the object transaction service and the security service to maintain transactional integrity and access control. Just as IDL defines the interface of a distributed object, a new persistent state definition language (PSDL) defines a portable distributed object schema. Object transaction service

- **Object transaction service**: (OTS) allows CORBA objects to operate within the framework of distributed transactions. The OTS specification was one of the first CORBA service specifications and has undergone recent changes to increase its flexibility and resolve various implementation issues. A transaction describes a collection of interactions where multiple users can access and/or modify data, and data integrity is guaranteed.

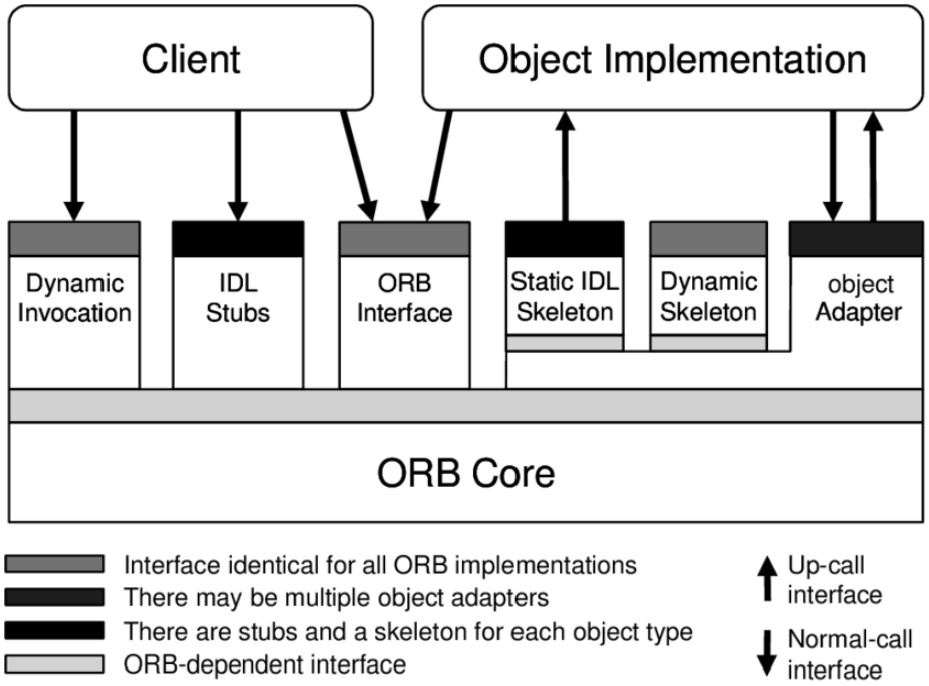

Fig.2: the architecture of CORBA (detailed)

3. Advantages and disadvantages of CORBA

++

Accessibility: One of the key benefits of CORBA is that it allows methods to be executed on a remote object as if it were local. This means that developers can invoke methods on objects located anywhere on the network without having to worry about the underlying communication details.

- **Maturity**: CORBA is a mature technology with a long history of use in enterprise environments. As a result, there is a wealth of support and tools available for developers, making it easier to implement and maintain CORBA-based systems.

- -

- **Complexity**: Despite its benefits, CORBA is known for its complexity. The learning curve can be steep for new developers, and setting up a CORBA environment requires a good understanding of its architecture and components.
- **Performance Overhead**: Using CORBA can introduce additional download time and performance overhead due to the need to transmit object requests and responses over the network. This can be particularly noticeable in systems with high latency or limited bandwidth.

  In summary, while CORBA offers powerful features for distributed object communication, its complexity and performance overhead are factors that need to be considered when choosing it as a solution for component-based development.

4. Conclusion

- CORBA is a component-based development technology for interoperable distributed objects.
- It allows information exchange across diverse hardware, languages, and operating systems.
- The main goal of CORBA is to create a market for component-based software that ensure interoperability regardless of hardware, language, or operating system.
- The architecture of CORBA is based on:
  - Object Request Broker (ORB): The heart of CORBA, transmits requests and responses.
  - Interface Definition Language (IDL): Defines the interface of a distributed object.
  - Stub: Client-side interface for making remote invocations.
  - Skeleton: Server-side interface for receiving remote invocations.
  - Internet Inter-ORB Protocol (IIOP): Standard communication protocol for CORBA.
- CORBA is easy to invoke methods on remote objects, it is well-established technology with extensive support and tools.
- But suffers from some Complexity: Steep learning curve and requires understanding of the architecture, and Performance Overhead: Additional download time and performance overhead.