Distributed Network Programming

Prativa Nyaupane

Review

- Architecture of RMI
- Creating and Executing RMI applications

Outline

- TCP
- UDP
- IP Address
- Ports
- Socket Programming using TCP and UDP
- Working with URLs and URL Connection Class
- Email Handling using Java Mail API
- Architecture of RMI
- Creating and Executing RMI applications
- Architecture of CORBA
- RMI vs CORBA
- IDL and Simple CORBA Program

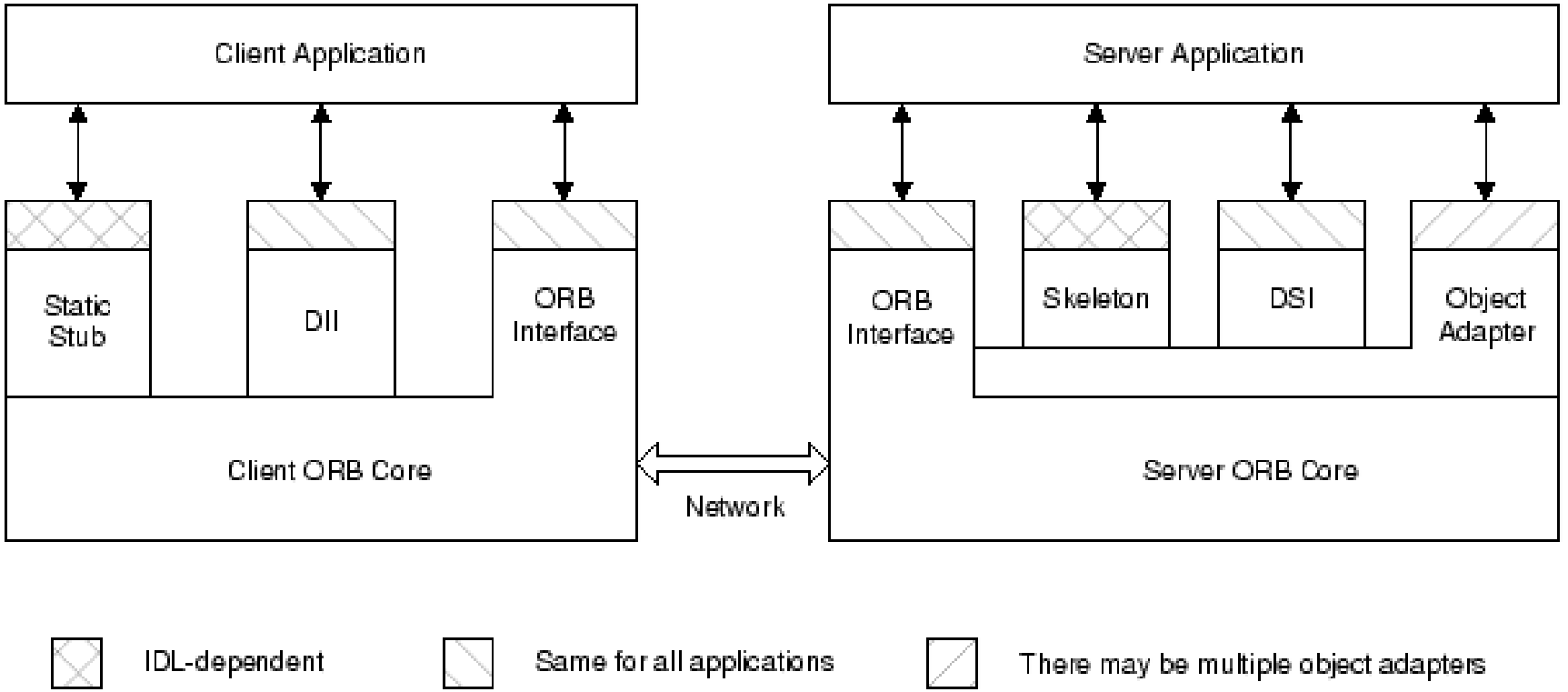
CORBA

- The **Common Object Request Broker Architecture(CORBA)** is a standard developed by the Object Management Group(OMG) to provide interoperability among distributed objects.
- **CORBA** is a standard for distributed object-oriented programming.
- **CORBA** allows objects written in different programming languages,operating systems and computing hardware to communicate with each other across a network.
- **CORBA** is an example of the distributed object paradigm.

CORBA Architecture

- At its core, the CORBA architecture for distributed objects shares many features with the architecture used by Java RMI.
- A description of a remote object is used to generate a client stub interface and a server skeleton interface for the object.
- A client application invokes methods on a remote object using client stub.
- The method request is transmitted through the underlying infrastructure to the remote host, where the server skeleton for the object is asked to invoke the method on the object itself.
- Any data resulting from the method call is transmitted back to the client by the communication infrastructure.
- The similarities end here.

CORBA Architecture Contd...



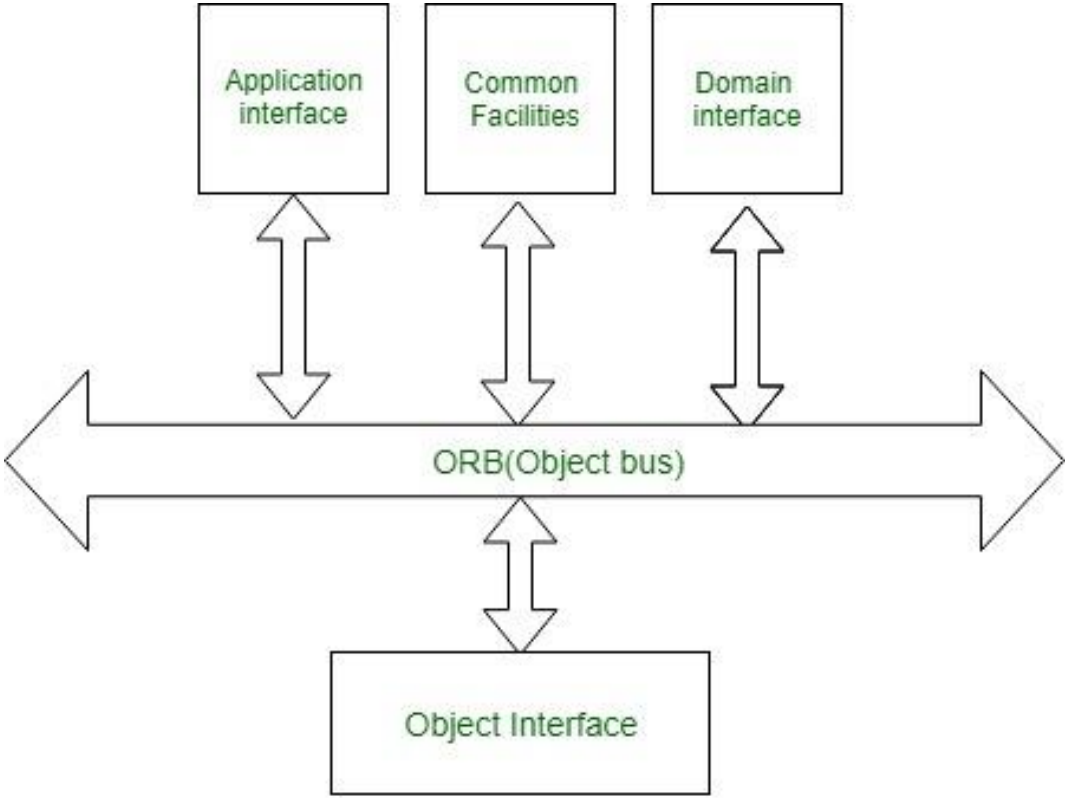
CORBA Architecture Contd...

- The **client** can choose to make requests either using **static stubs**(generated from either the IDL definition) or using the **Dynamic Invocation Interface(DII)**. Either way, the client directs the request to the ORB core linked to its process.
- The **Object Request Broker(ORB)** is the core component of CORBA architecture. It is responsible for handling communication between objects and for locating and activating remote objects.
- The **client ORB** core transmits the request to the ORB core linked with the server application.
- The **server ORB** core dispatches the request to the object adapter that created the target object.

CORBA Architecture Contd...

- The **object adapter** further dispatches the request to the servant that is implementing the target object.
- **Like the client**, the server can choose between static and dynamic dispatching mechanisms for its servants. It can rely on **static skeletons** generated from the object's IDL definition, or its servants can use the **Dynamic Skeleton Interface(DSI)** .
- After the servant carries out the request, it returns its response to the client application.

Object Management Architecture



Object Management Architecture(OMA)

Working with CORBA(example)

- Using a CORBA implementation, a shopper will transparently invoke a way on server object, which may air a similar machine across a network.
- The middleware takes the decision, associated is to blame for finding an object which will implement the request, passing it the parameters, invoking its methodology, and returning the results of the invocation.
- The shopper does not need to remember where the item is found, its programming language, its software package or other aspects that don’t seem to be a part of the associated object’s interface.

CORBA vs RMI

CORBA	RMI
CORBA was designed from the start to be a language-independent distributed object standard, so it is much more extensive and detailed in its specification than RMI is.	RMI is a Java-specific technology.
It uses Interface Definition Language to separate instance from implementation.	It uses Java interface for implementation.
CORBA objects are not garbage collected because it is language independent and some languages like C++ does not support garbage collection.	RMI objects are garbage collected automatically.
CORBA does not support code sharing mechanism.	RMI programs can download new classes from remote JVM's
CORBA passes objects by reference.	RMI passes object by remote reference of by value.
CORBA is a peer-to-peer system.	Java RMI is a server-centric model.

Interface Definition Language (IDL)

- Language Independence:
  - CORBA allows objects written in one language to communicate with objects implemented in a different language.
  - For example, objects in Java or Smalltalk can interact with objects written in C or COBOL.
- Specification Meta-Language:
  - Achieves language independence through a meta-language specification.
  - This meta-language defines the interfaces that objects present to the outside world.
- Object-Oriented Nature:
  - Objects in CORBA, like in any object-oriented system, have private data and private methods.
  - The interface, specifying public data and methods, is what the object shows externally

Interface Definition Language (IDL)

- Interface Definition Language (IDL):
  - IDL is the language used by CORBA to specify its objects.
  - It is not for writing procedural code; its sole purpose is to define data, methods, and exceptions.
- Compiler and Translation:
  - Each CORBA vendor provides a compiler that translates IDL specifications into a specific programming language.
  - For example, Oracle8i JServer uses the idl2java compiler from Inprise for Java translation.
- Vendor-Specific Tools:
  - The idl2java compiler transforms IDL interface specifications into Java classes.
  - Users can refer to the Oracle8i Java Tools Reference for more details on this tool.