CHAPTER 2. Introduction to Middleware Technologies

- What is Middleware?
- General Middleware
- Service Specific Middleware
- Client/Server Building blocks
- RPC
- Messaging
- Peer – to – Peer
- Java RMI.

What is Middleware?

- ❖ Middleware is software that runs between client and server processes. It is the "glue" between the client and server, which makes it possible for them to communicate to each other. Middleware is written in such a way that the user never notices it's presence.
- ❖ It delivers secure and transparant services to users.
- ❖ It is used most often to support complex, distributed applications.
- ❖ It includes web servers, application servers, content management systems, and similar tools that support application development and delivery.
- ❖ Middleware is especially integral to modern information technology based on XML,SOAP, Web services, and service-oriented architectur

Types of middleware services

- Remote Data Access (RDA), which implements a RDA protocol for sending data manipulation language statements to an appropriate database server for processing and transporting the result back to the invoking process.
- Remote procedure calls (RPCs). RPC is used in most network operating system services.
- Message-oriented middleware (MOM). MOM can be used as a mechanism for storing and forwarding messages queuing.It can be used when client and server processes communicate asynchronously.
- Object Request Brokers(ORBs). A standard implementation of the ORB standard is CORBA. ORB makes it possible to invoke a remote object by allowing a source object to send a message to that remote object.
- Distributed transaction processing (DTP). This type of mechanism use execution semantics to interact between the client and the server.

- It starts with the API set on the client side that is used to invoke a service, and it covers the transmission of the request over the network and the resulting response.
- Middleware divided into two broad classes:
  (a) General Middleware
  (b) Service-Specific Middleware

General Middleware

- It is the substrate for most client/server interactions
- It includes the communication stacks, distributed directories, authentication stacks, distributed directories, authentication services, network time, remote procedure calls, and queuing services.
- Products that fall into the general middleware category include DCE, ONC+, NetWare, NamedPipes, LAN Server, LAN Manager, Vines, TCP/IP, APPC and NetBIOS.
- DCE is an example of Generic middleware.

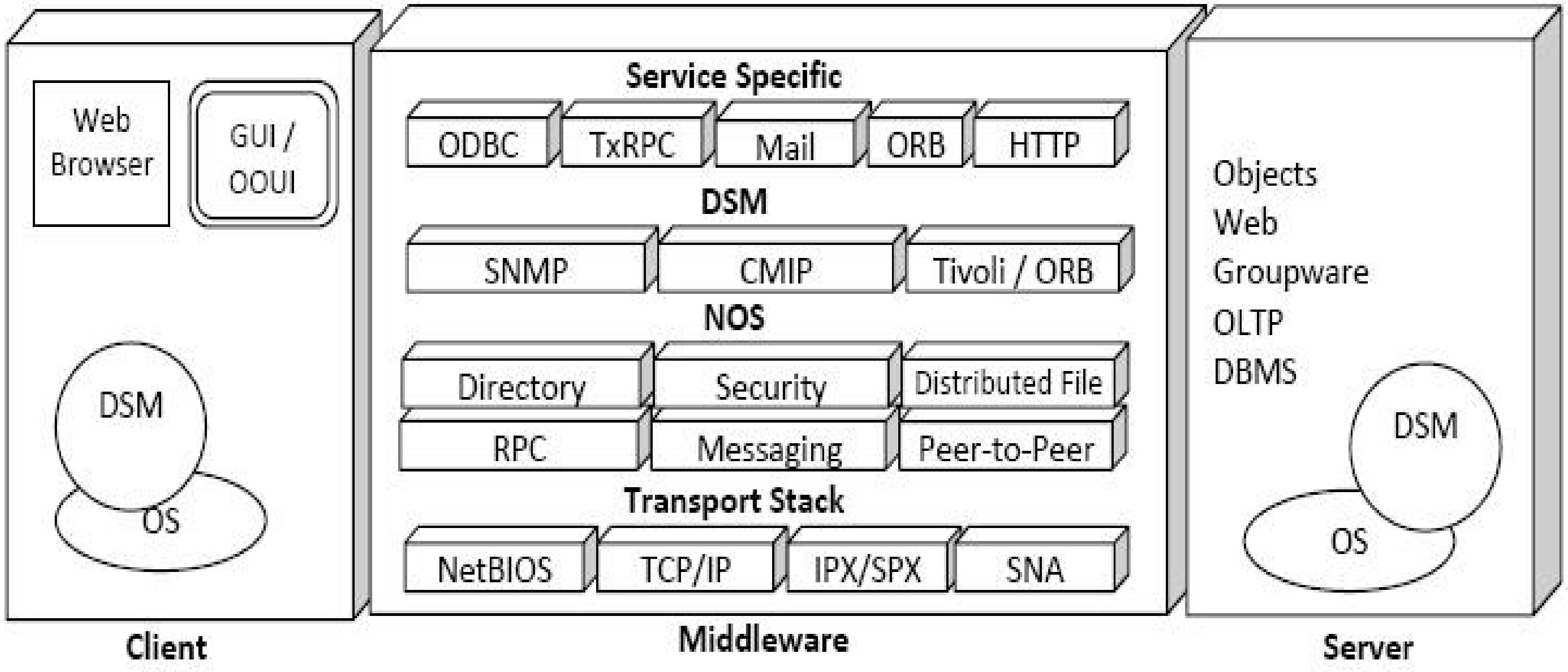
Service-Specific Middleware

- It is need to accomplish a particular client/server type of service.
- This includes
  - **Database middleware** such as ODBC, DRDA, EDA/SQL, SAG/CLI and Oracle Glue.
  - **OLTP-specific middleware** such as Tuxedo's ATMI and /WS, Encina's Transactional RPC, and X/Open's TxRPC and XATMI
  - **Groupware-specific middleware** such as MAPI, VIM, VIC, SMTP and Lotus Notes Calls
  - **Object-specific middleware** such as OMG's CORBA and Microsoft's Network OLE (or DCOM)
  - **Internet-specific middleware** such as HTTP, S-HTTP and SSL
  - **System Management-specific middleware** such as SNMP, CMIP and ORBs.

Client/Server Middleware

The Building Blocks of Client/Server are

1).Client
2).MiddleWare(/)"slash",which ties the client to the server
3).Server



Client/Server Building blocks

1) The Client Building Block

- Runs the client side of the application
- It runs on the OS that provides a GUI or an OOUI and that can access distributed services, wherever they may be.
- The client also runs a component of the Distributed System Management (DSM) element.

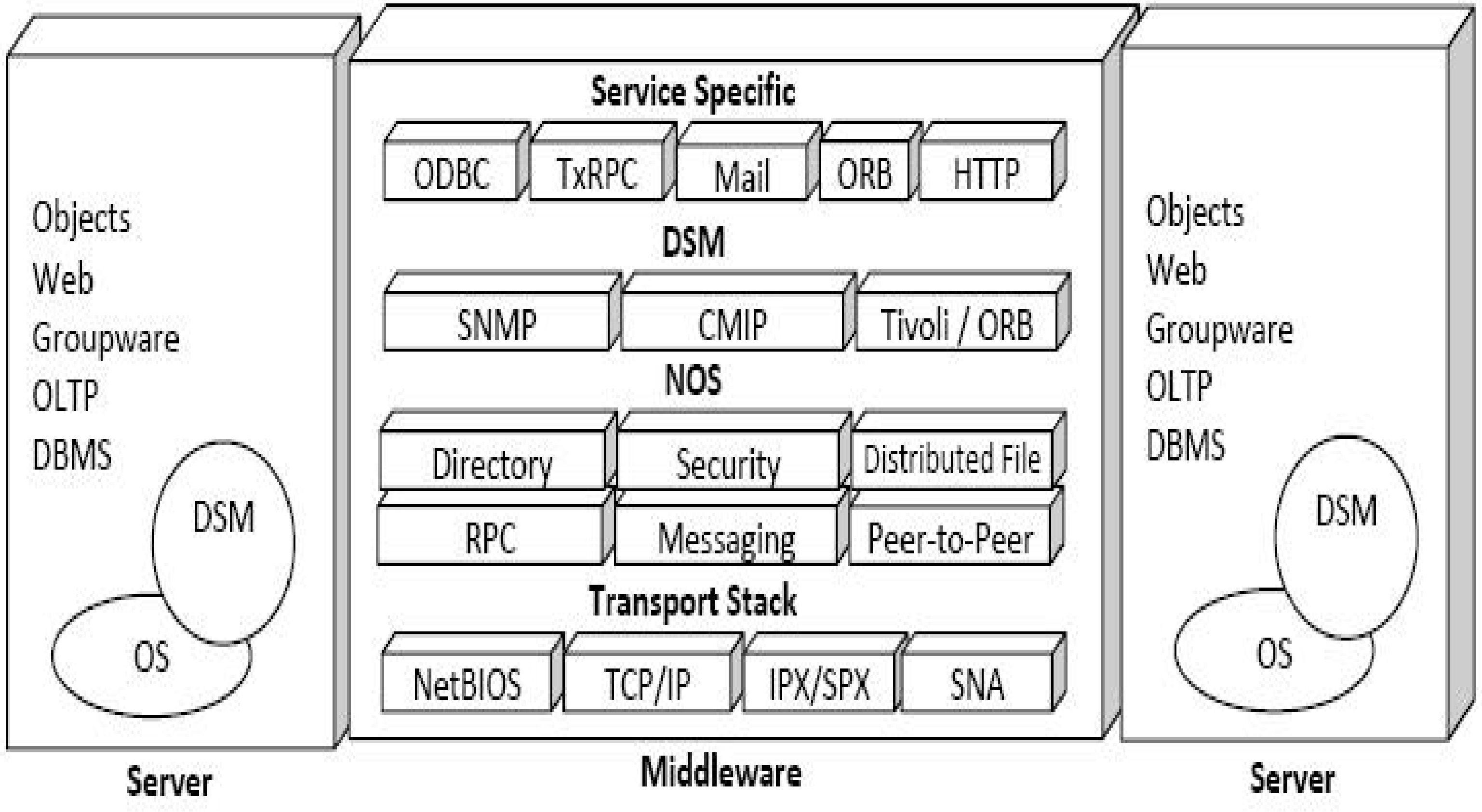
2) The Server Building Block

- Runs the server side of the application
- The server application typically runs on top of some shrink-wrapped server software package.
- The five contending server platforms for creating the next generation of client/server applications are SQL database severs, TP Monitors, groupware servers, Object servers and the Web server.
- The server side depends on the OS to interface with the middleware building block.
- The server also runs DSM component
- It may be a simple agent or a shared object database etc.

3)The Middleware Building Block

- Runs on both the client and server sides of an application
- This broken into three category
  - Transport Stacks
  - NOS
  - Service-specific middleware
- Middleware is the nervous system of the client/server infrastructure

Server-to-server Middleware



Server-to-server Middleware

- Server-to-server interactions are usually client/server in nature – servers are clients to other servers.
- However, some server-to-server interactions require specialized server middleware. For example, Two-Phase commit protocol may be used to coordinate a transaction that executes on multiple servers.
- Servers on mail backbone will use special server-to-server middleware for doing store-and-forward type messaging.
- But most modern software follows the client/server paradigm.

Peer to Peer Communication

In this model client and server both side of link uses same protocol interface for network communication Any computer on network can initiate communication with any computer Each computer is responsible for making its own recourses available

The Example of peer to peer communication is Sockets

The Socket is a communication end points which enable programmer to write a network Application

Peer to Peer Communication

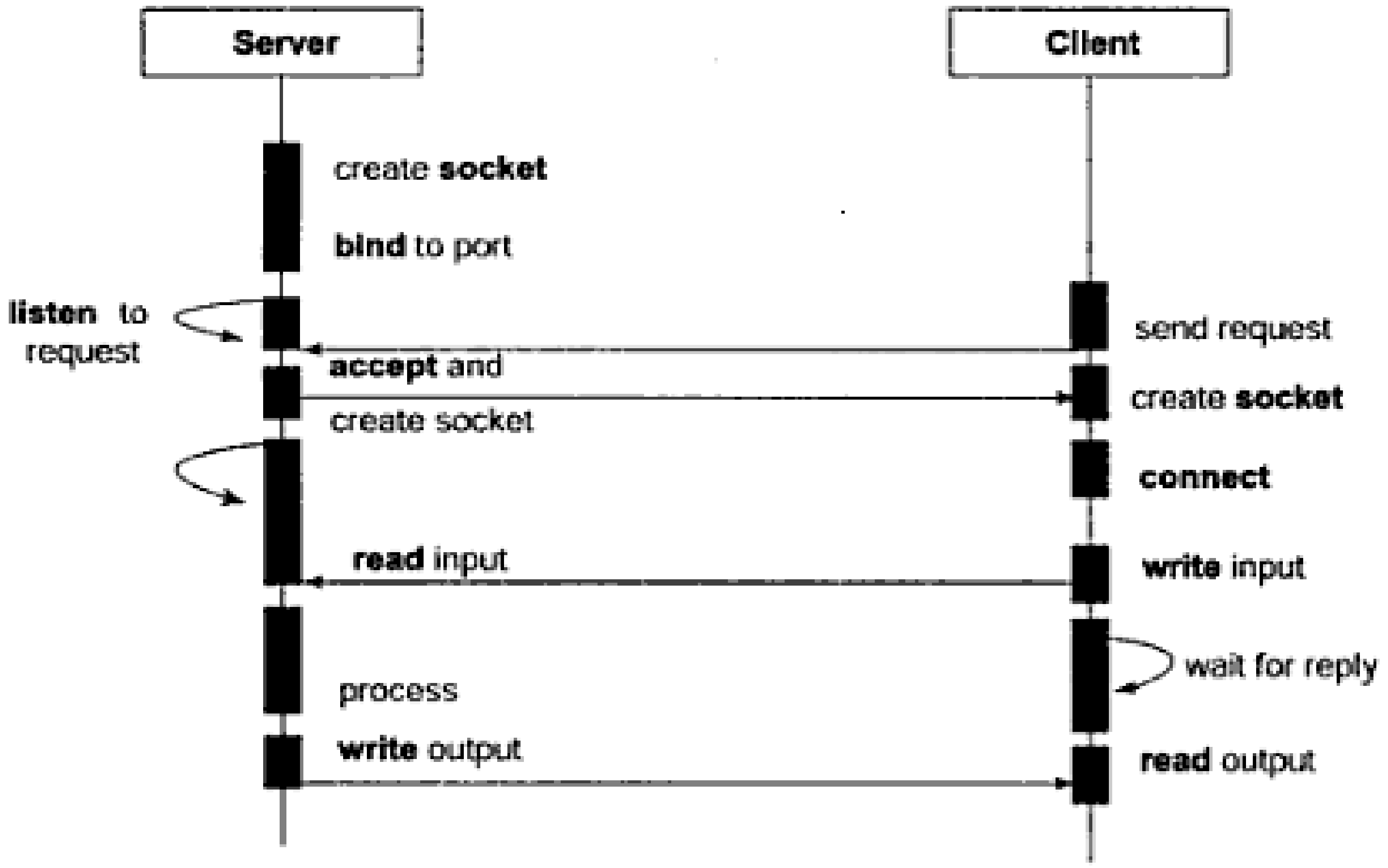


Figure 1-9 Client–server interaction in connection-based protocol.

REMOTE PROCEDURE CALL

DEFINATION

When a process on machine *A* calls a procedure on machine *B*, the calling process on *A* is suspended, and execution of the called procedure takes place on *B*. Information can be transported from the caller to the callee in the parameters and can come back in the procedure result.No message passing at all is visible to the programmer. This method is known as **Remote procedure call** (**RPC**) .

- **Remote procedure call** (**RPC**) is an Inter-process communication technology that allows a computer program to cause a subroutine or procedure to execute in another address space (commonly on another computer on a shared network) without the programmer explicitly coding the details for this remote interaction

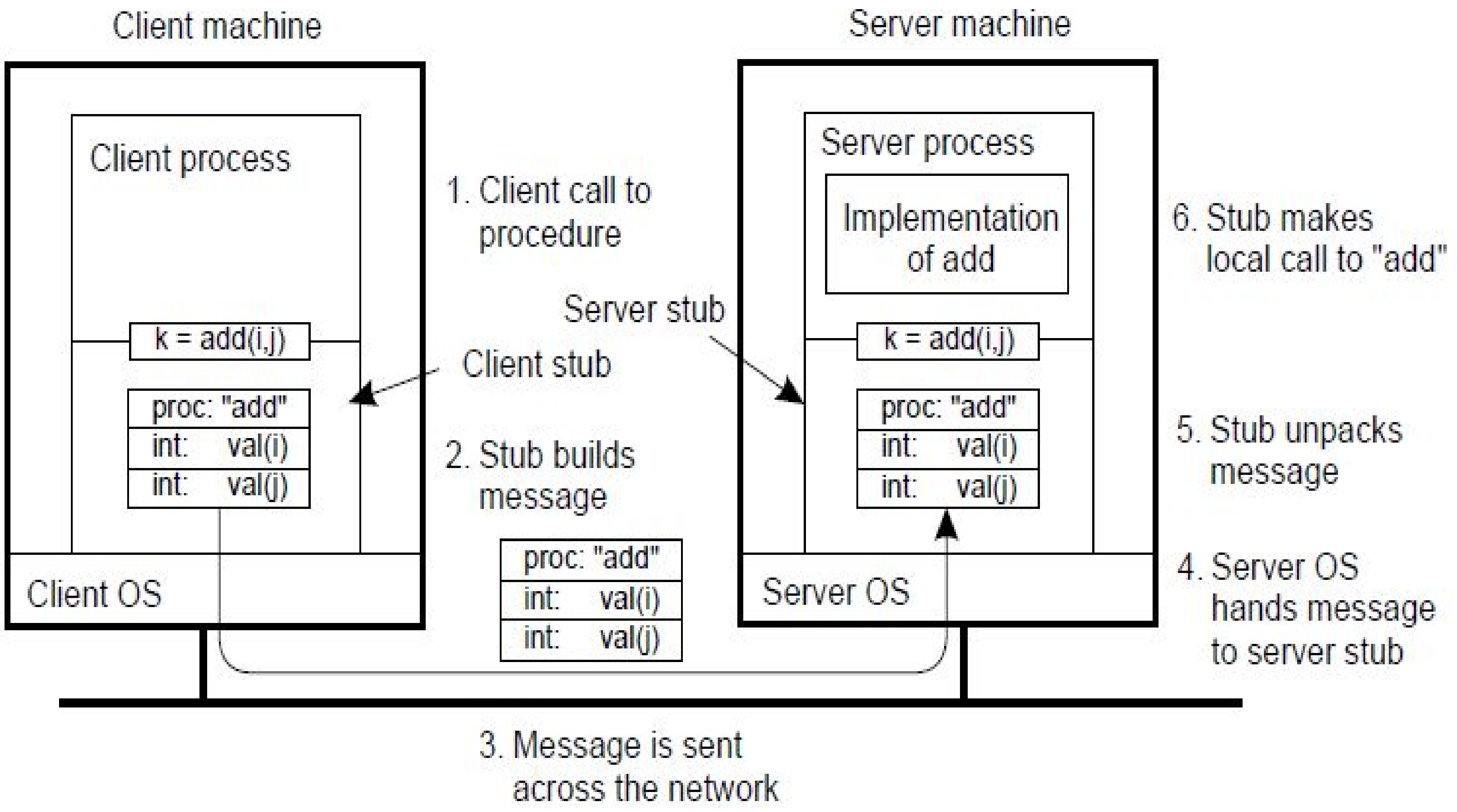


Figure 2-3. The steps involved in a doing a remote computation through RPC.

Remote Procedure Call

Remote procedure call occurs in the following steps:

1. The client procedure calls the client stub in the normal way.
2. The client stub builds a message and calls the local operating system.
3. The client's OS sends the message to the remote OS.
4. The remote OS gives the message to the server stub.
5. The server stub unpacks the parameters and calls the server.
6. The server does the work and returns the result to the stub.
7. The server stub packs it in a message and calls its local OS.
8. The server's OS sends the message to the client's OS.
9. The client's OS gives the message to the client stub.
10. The stub unpacks the result and returns to the client.

Remote Procedure Call

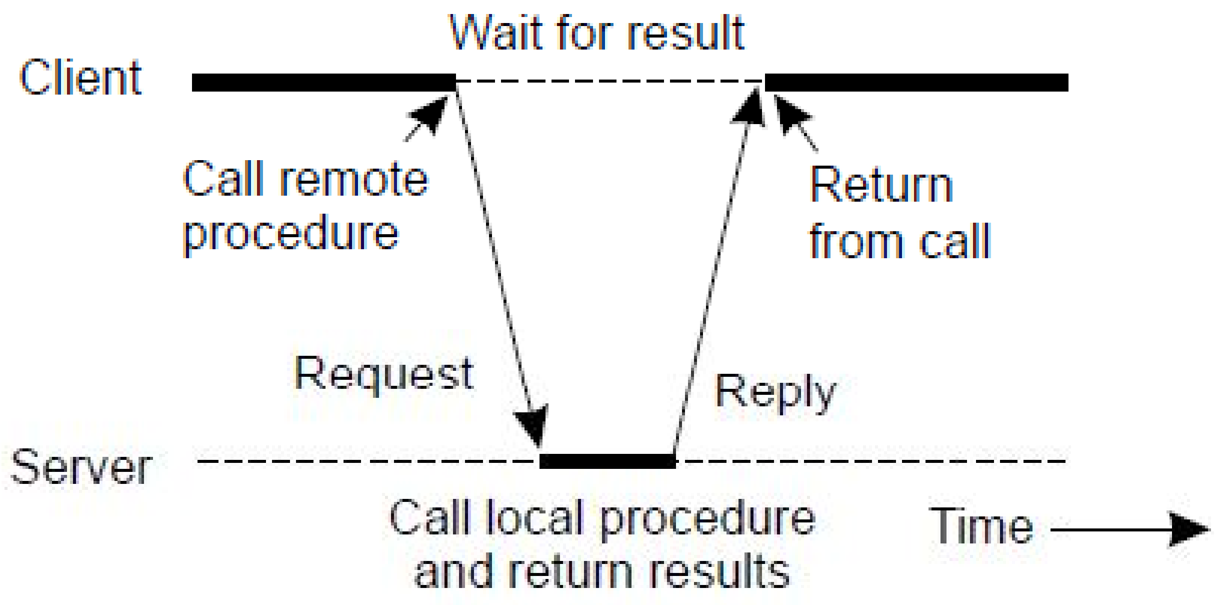


Figure 2-2. Principle of RPC between a client and server program.

RPC MODEL

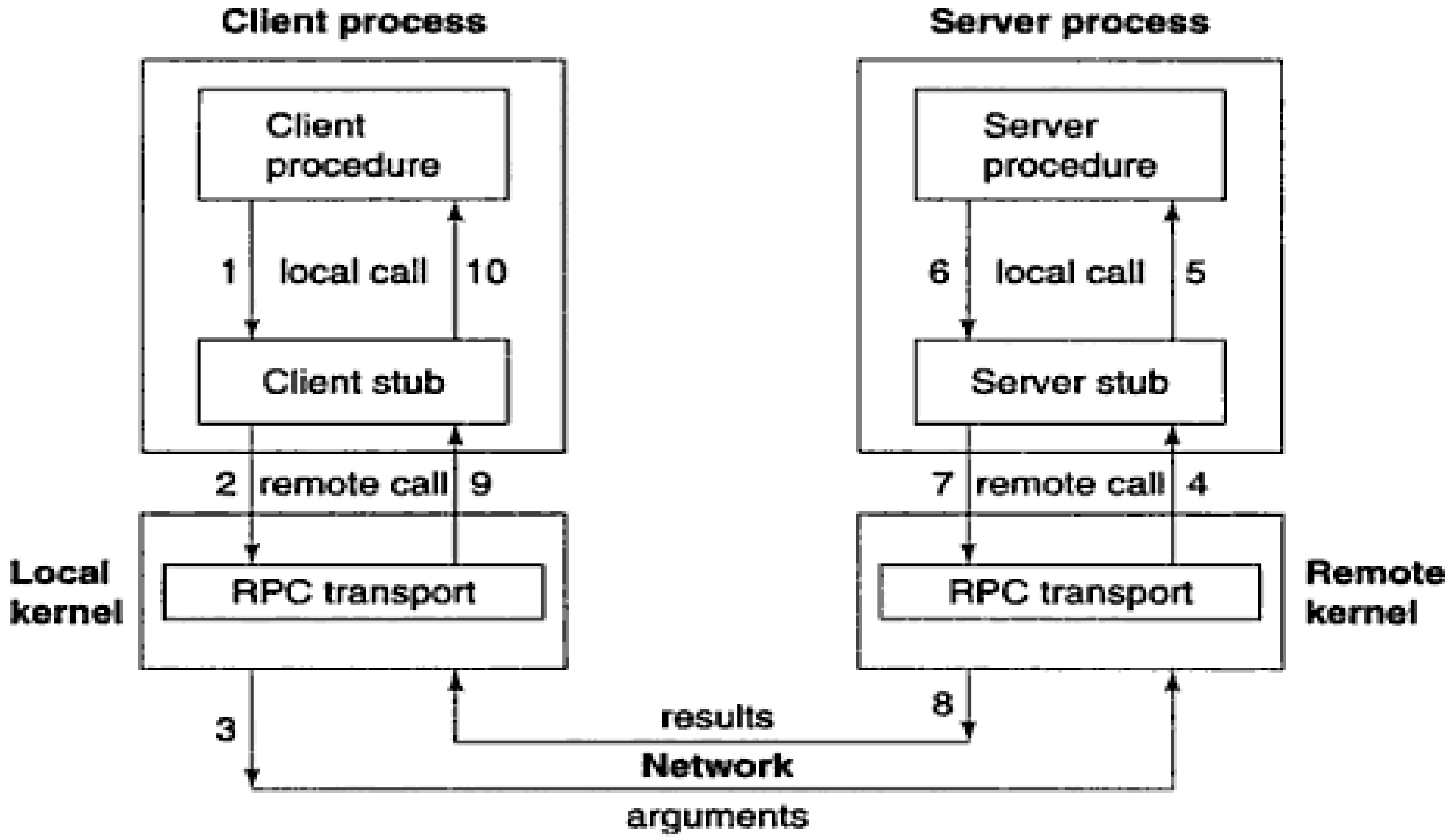


Figure 1-11 RPC model.

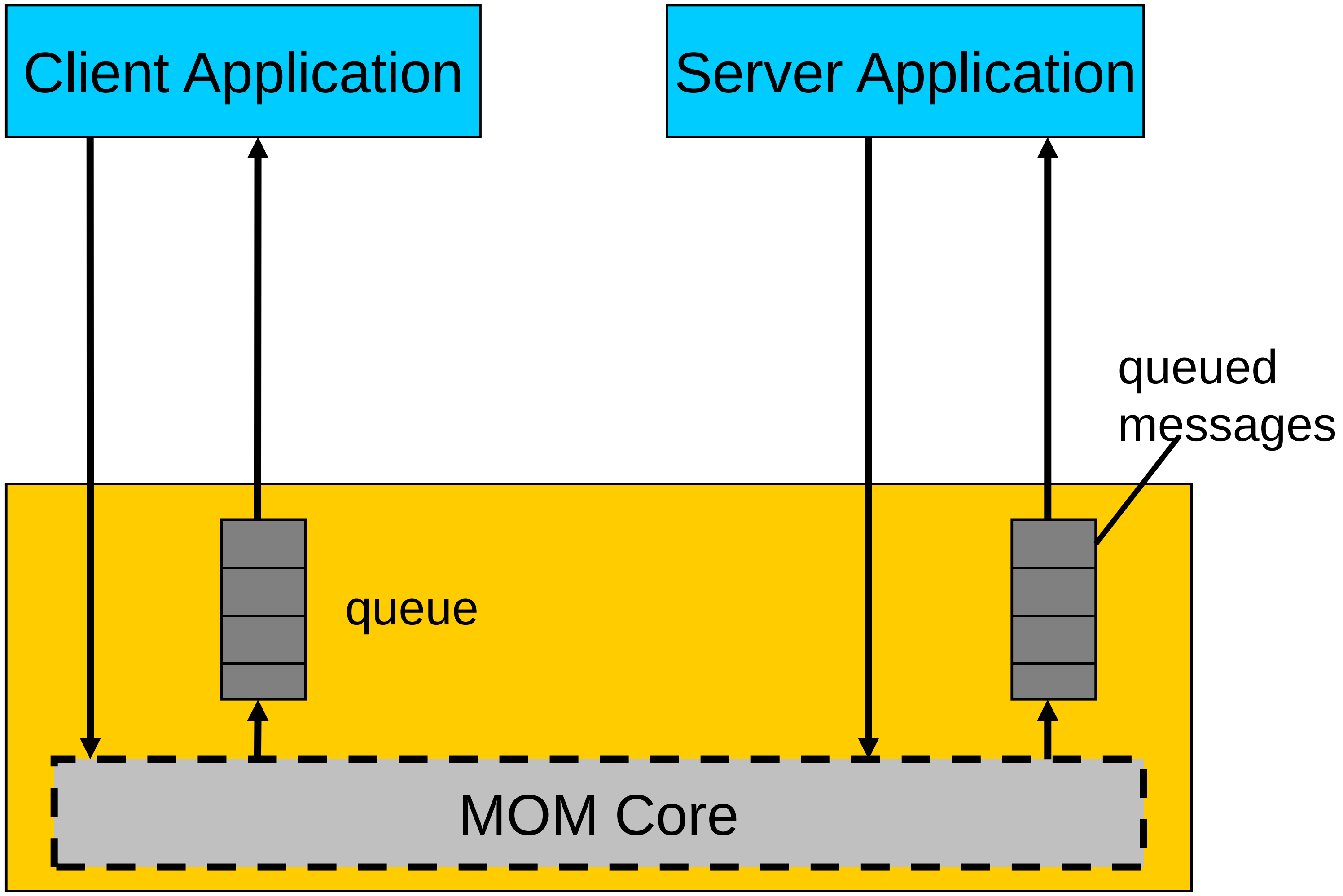
Message-oriented middleware

Message Oriented Middleware or MOM-based middleware, which allows distributed applications to communicate and exchange data by sending and receiving messages. **Message-oriented middleware (MOM)** is infrastructure focused on sending and receiving messages that increases the interoperability, portability, and flexibility of an application by allowing the application to be distributed over heterogeneous platforms. It reduces the complexity of developing applications that span multiple operating systems and network protocols by insulating the application developer from the details of the various operating system and network interfaces. API's that extend across diverse platforms and networks are typically provided by MOM

Message-Oriented middleware

- Asynchronous forms of interaction
- Communication by exchanging messages
- More dynamic than RPC and Sockets

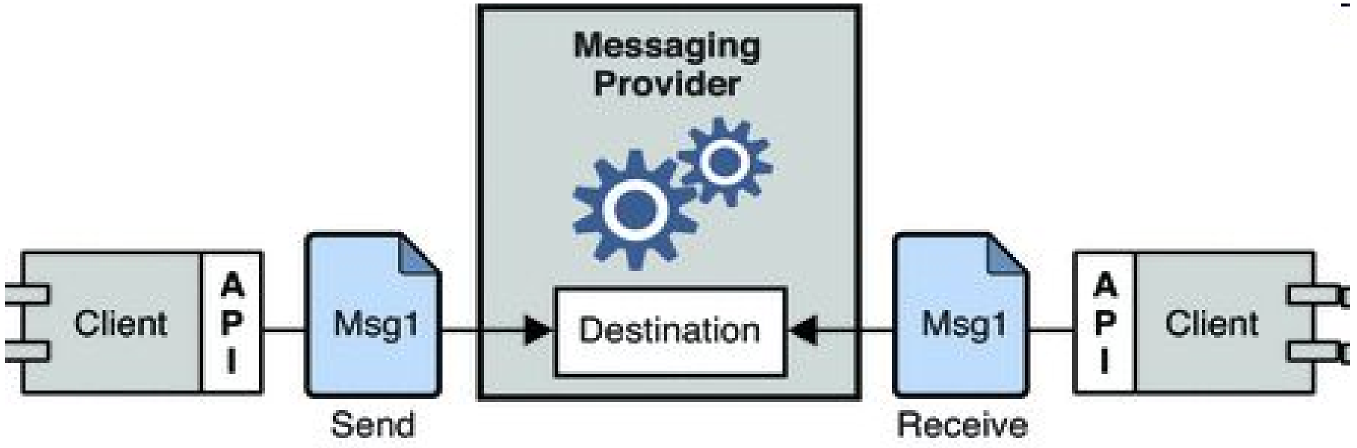
Message-Queues



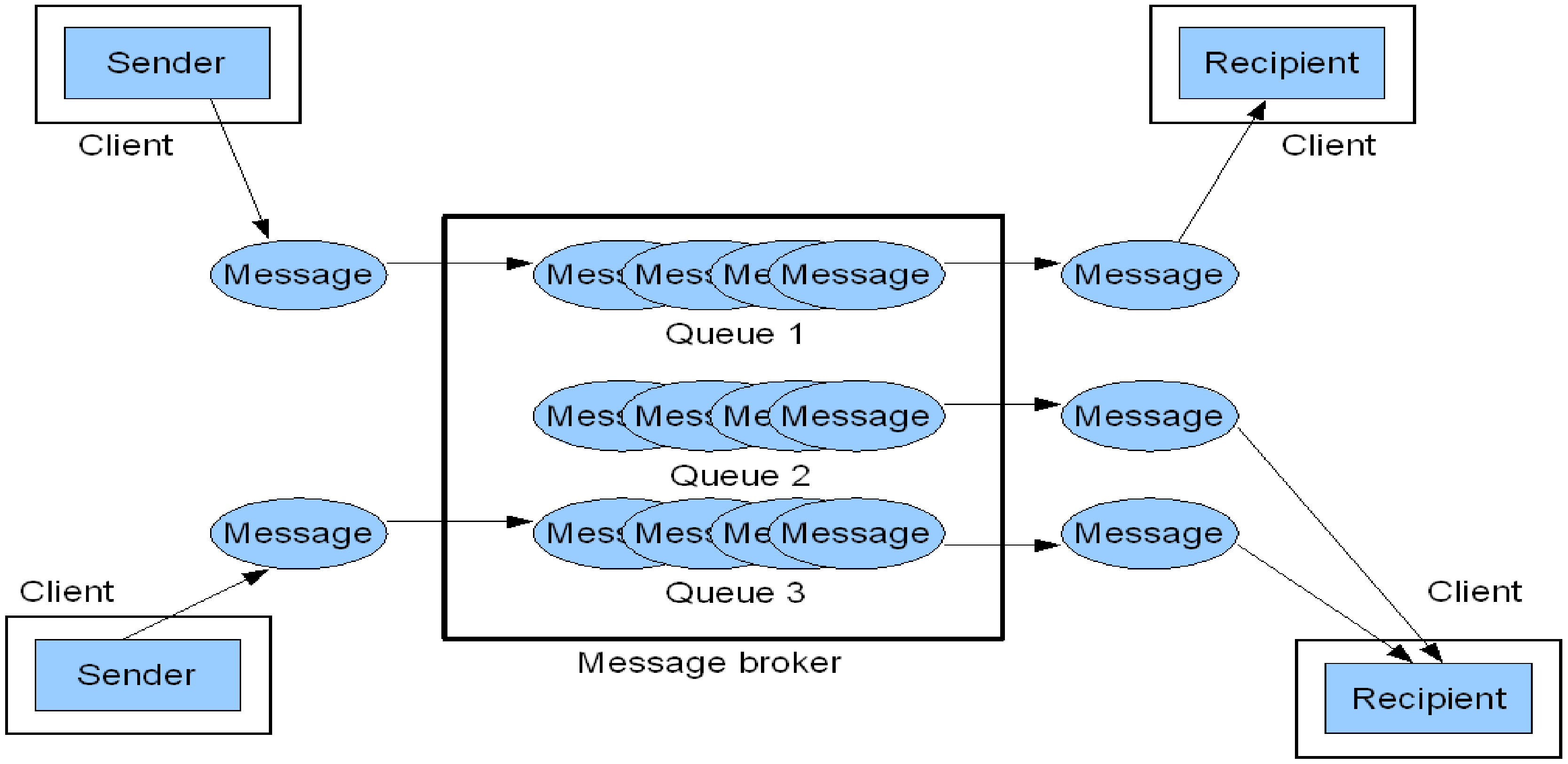
Message-Oriented middleware

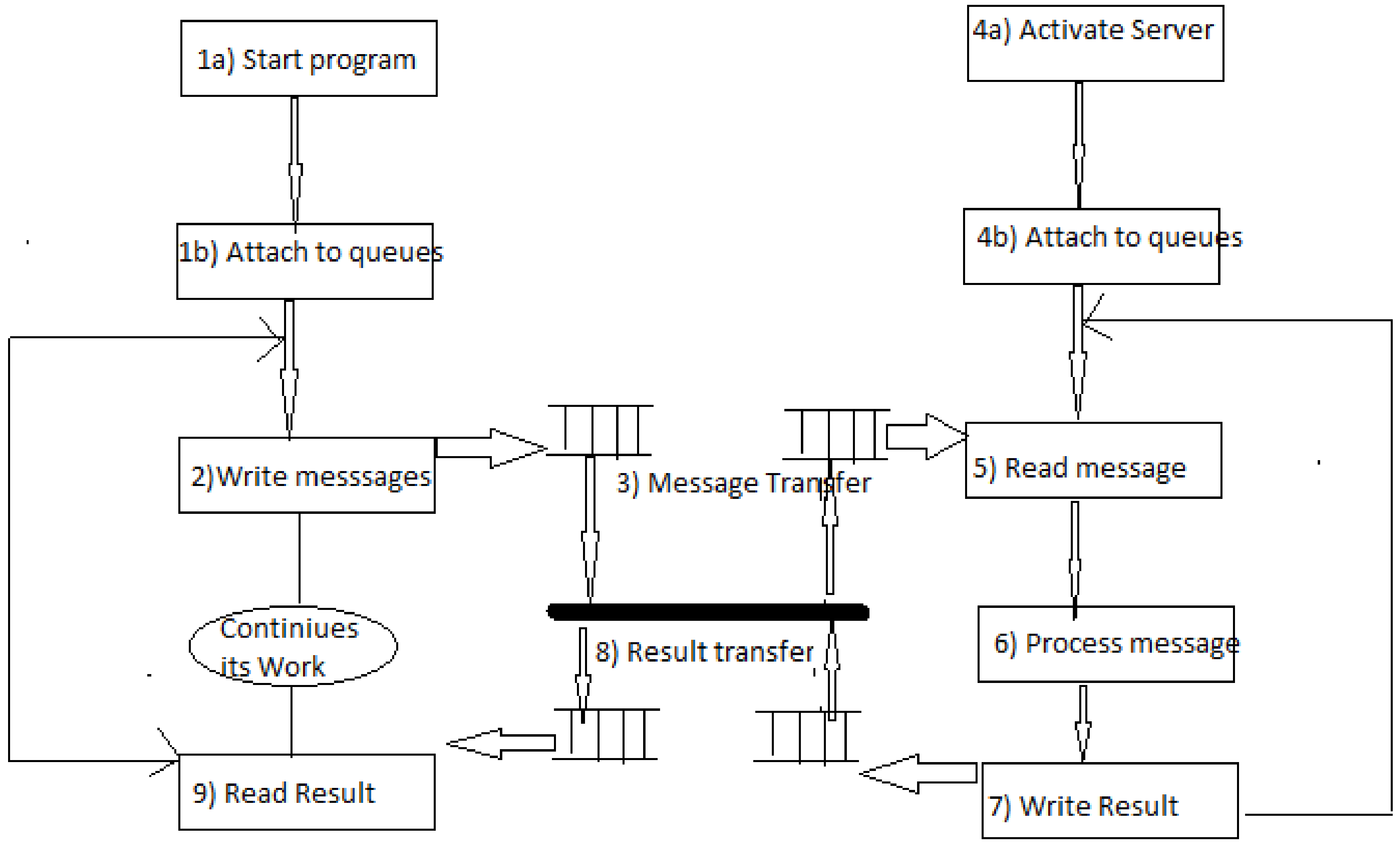
Using a MOM system, a client makes an API call to send a message to a destination managed by the provider. The call invokes provider services to route and deliver the message. Once it has sent the message, the client can continue to do other work, confident that the provider retains the message until a receiving client retrieves it. The message-based model, coupled with the mediation of the provider, makes it possible to create a system of loosely-coupled components. Such a system can continue to function reliably, without downtime, even when individual components or connections fail.

The basic elements of a MOM system are clients, messages, and the MOM provider, which includes an API and administrative tools. The MOM provider uses different architectures to route and deliver messages



Message-Oriented middleware





Message Queue Communication Protocol

Message-Oriented middleware

Advantages

Most MOM systems provide persistent storage to back up the message transfer medium. This means that the sender and receiver do not need to connect to the network at the same time (asynchronous delivery).

MOM delivers another important advantage through its ability to route messages within the middleware layer itself.

Message-Oriented middleware

Disadvantages

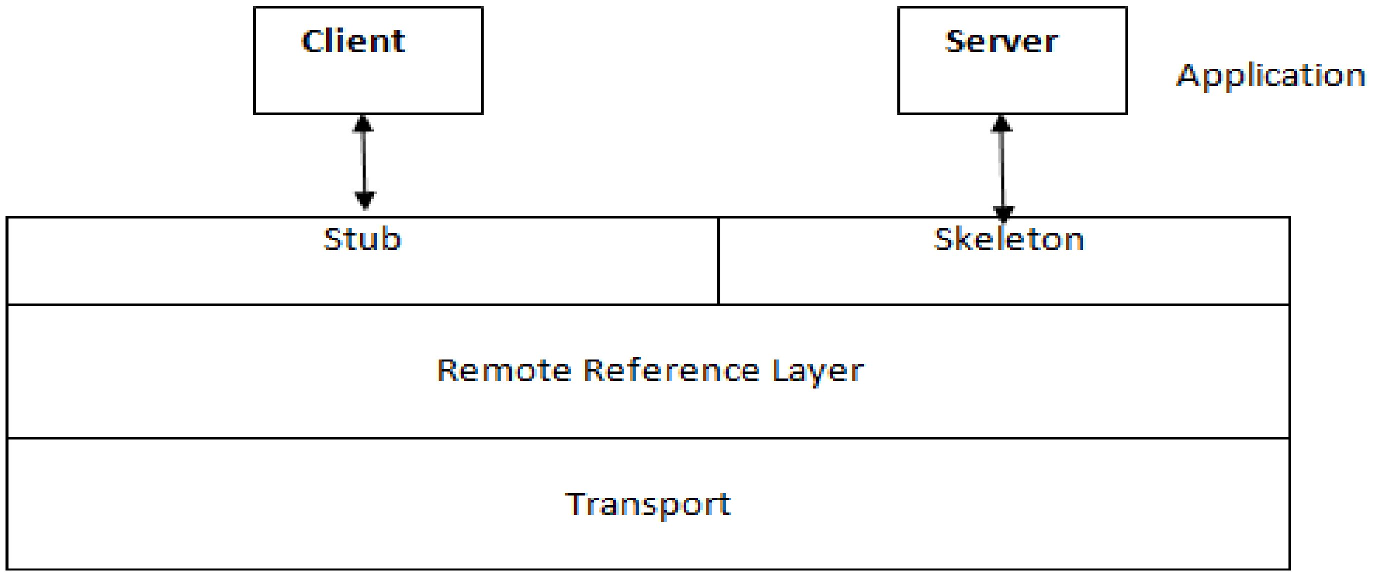
The primary disadvantage of many message oriented middleware systems is that they require an extra component in the architecture, the message transfer agent (Message broker). As with any system, adding another component can lead to reductions in performance and reliability, and can also make the system as a whole more difficult and expensive to maintain.

Lack of standards

The lack of standards governing the use of message oriented middleware has caused problems. All the major vendors have their own implementations, each with its own application programming interface (API) and management tools.

Remote Method Invocation

For Theory Refer Class Notes



Architecture of RMI

Remote Method Invocation

Refer Class Notes for Diagram of Rmi Process & its Theory.

- ANY QUESTION

T H A N K

Y O U