MURANG'A UNIVERSITY OF TECHNOLOGY
DEPARTMENT OF IT

SIT404: CLIENT SERVER SYSTEMS
CREDIT HOURS: 3 HOURS

COURSE NOTES – 2020/2021

INTERFACES

Recall:
A client/server environment is populated by clients and servers.
The client machines are generally single-user PCs or workstations that provide a highly user-friendly interface to the end user.
Each server provides a set of shared user services to the clients.
Server enables many clients to share access to the same database.
Users, application, and resources are distributed in response to business requirements and linked by a single LAN or WAN or by an internet of networks.

INTERFACES CONT'D

The program interface allows communication between the two programs.
In the dedicated server configuration, communication between the user and server processes occurs using different mechanisms:
=> If the system is configured so that the user process and the dedicated server process are run by the same computer, the program interface uses the host operating system's inter-process communication mechanism to perform its job.

INTERFACES CONT'D

=> If the user process and the dedicated server process are executed by different computers, the program interface also encompasses the communication mechanisms, such as the network software and SQL*Net, between the programs.

*SQL *Net enables both client-server and server-server communications across any network.*

*With SQL*Net, databases and their applications can reside on different computers and communicate as peer applications*

These communications links are operating system- and installation-dependent

PROTOCOLS

The instructions and conventions needed for successful communication is known as a protocol.

To ease the task of communicating and provide a degree of flexibility, network protocols are generally organized in layers. This allows you to replace a layer of the protocol without having to replace the surrounding layers.

It saves higher-level software from having to bother with formatting an Ethernet packet

PROTOCOLS CONT'D

A communications protocol provides a structure for requests between client and server in a network.

For example, the Web browser in the user's computer (the client) employs the HTTP protocol to request information from a website on a server

The Internet protocol suite is the set of communications protocols that implement the protocol stack on which the Internet and most commercial networks run.

PROTOCOLS CONT'D

i) TCP/IP

It has also been referred to as the TCP/IP protocol suite, which is named after two of the most important protocols in it:

=> the Transmission Control Protocol (TCP) and

=> the Internet Protocol (IP).

TCP/IP is referred as protocol suite because it contains many different protocols and therefore many different ways for computers to talk to each other.

TCP/IP is not the only protocol suite, although TCP/IP has gained wide acceptance and is commonly used

PROTOCOLS CONT'D

TCP/IP also defines conventions by connecting different networks, and routing traffic through routers, bridges, and other types of connections.
The TCP/IP suite is result of a Defense Advanced Research Projects Agency (DARPA) research project about network connectivity, and its availability has made it the most commonly installed network software.
OSI was developed as theoretical model, while TCP/IP was more practical.
All the layers are roughly corresponding to the OSI model

PROTOCOLS CONT'D

Clients typically communicate with servers by using the TCP/IP protocol suite.

TCP is a connection-oriented protocol, which means a connection is established and maintained until the application programs at each end have finished exchanging messages.

It determines how to break application data into packets that networks can deliver, sends packets to and accepts packets from the network layer, manages flow control and handles retransmission of dropped or garbled packets as well as acknowledgement of all packets that arrive.

PROTOCOLS CONT'D

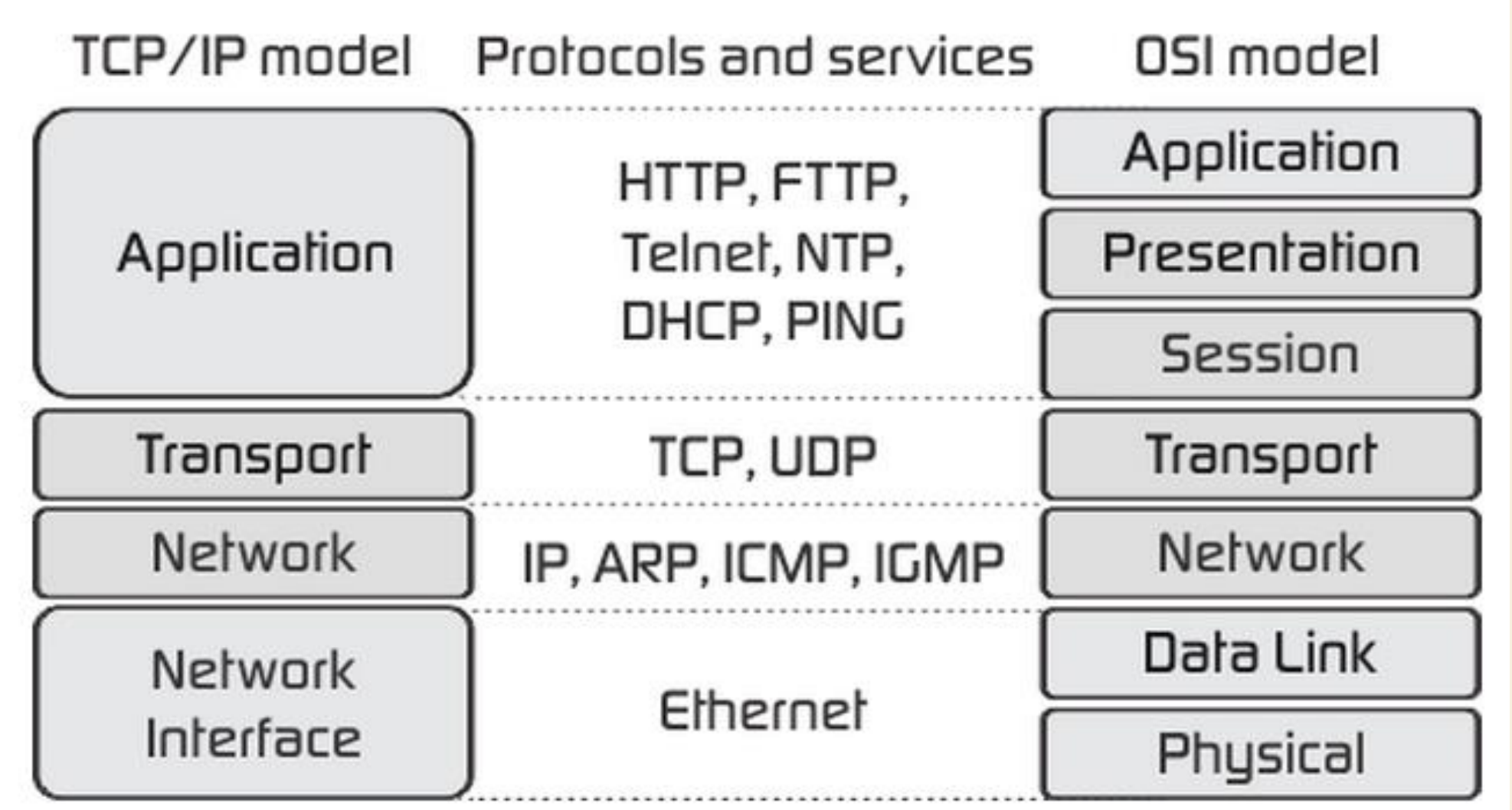
In the Open Systems Interconnection (OSI) communication model, TCP covers parts of Layer 4, the Transport Layer, and parts of Layer 5, the Session Layer.

In contrast, IP is a connectionless protocol, which means that there is no continuing connection between the endpoints that are communicating.

Each packet that travels through the Internet is treated as an independent unit of data without any relation to any other unit of data. (The reason the packets do get put in the right order is because of TCP.)

In the Open Systems Interconnection (OSI) communication model, IP is in layer 3, the Networking Layer.

PROTOCOLS CONT'D



PROTOCOLS CONT'D

ii) SNMP

The Simple Network Management Protocol (SNMP) forms part of the internet protocol suite as defined by the Internet Engineering Task Force (IETF).

SNMP is used in network management systems to monitor network-attached devices for conditions that warrant administrative attention.

It consists of a set of standards for network management, including an Application Layer protocol, a database schema, and a set of data objects.

PROTOCOLS CONT'D

SNMP exposes management data in the form of variables on the managed systems, which describe the system configuration.
These variables can then be queried (and sometimes set) by managing applications.
In typical SNMP usage, there are a number of systems to be managed, and one or more systems managing them.
A software component called an agent runs on each managed system and reports information via SNMP to the managing systems

PROTOCOLS CONT'D

An SNMP-managed network consists of three basic key components:

=> Managed devices

=> Agents

=> Network-Management Systems (NMSs)

A managed device is a network node that contains an SNMP agent and that resides on a managed network Managed devices collect and store management information and make this information available to NMSs using SNMP.

PROTOCOLS CONT'D

Managed devices, sometimes called network elements, can be any type of device including, but not limited to, routers and access servers, switches and bridges, hubs, IP telephones, computer hosts, or printers.

An agent is a network-management software module that resides in a managed device.

An agent has local knowledge of management information and translates that information into a form compatible with SNMP.

PROTOCOLS CONT'D

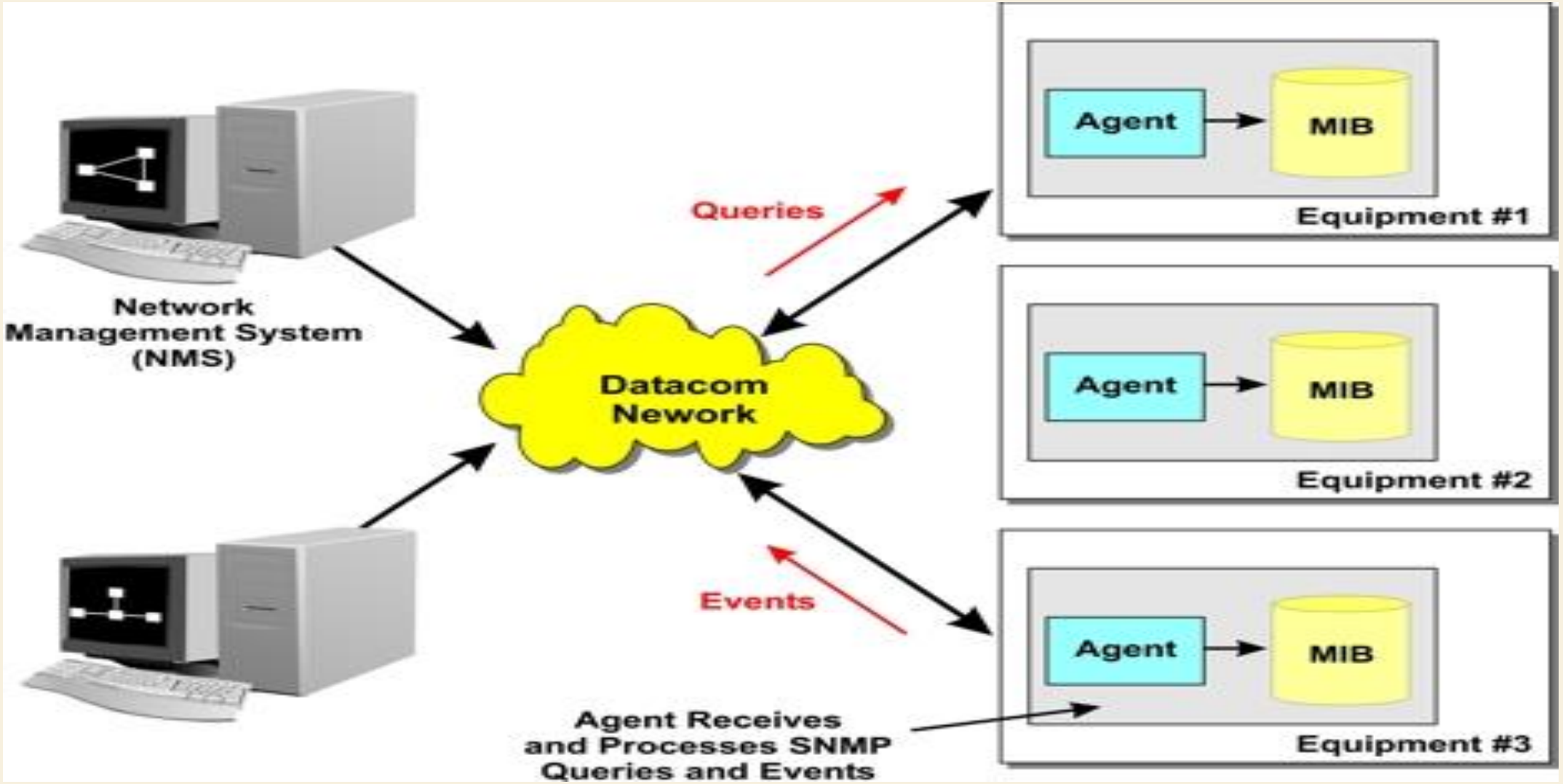
An NMS executes applications that monitor and control managed devices.

NMSs provide the bulk of the processing and memory resources required for network management.

One or more NMSs may exist on any managed network

A managed network is a type of communication network that is built, operated, secured and managed by a third-party service provider. A managed network is an outsourced network that provides some or all the network solutions required by an organization

PROTOCOLS CONT'D



PROTOCOLS CONT'D

iii) NFS

Network File System (NFS) is a network file system protocol originally developed by Sun Microsystems in 1984, allowing a user on a client computer to access files over a network as easily as if the network devices were attached to its local disks.

NFS, like many other protocols, builds on the Open Network Computing Remote Procedure Call (ONC RPC) system. Assuming a Unix-style scenario in which one machine (the client) requires access data, stored on another machine (the NFS server).

PROTOCOLS CONT'D

The server implements NFS daemon processes (running by default as NFSD) in order to make its data generically available to clients.

The server administrator determines what to make available, exporting the names and parameters of directories (typically using the/etc./exports configuration file and the exports command).

The server security-administration ensures that it can recognize and approve validated clients.

The server network configuration ensures that appropriate clients can negotiate with it through any firewall system.

PROTOCOLS CONT'D

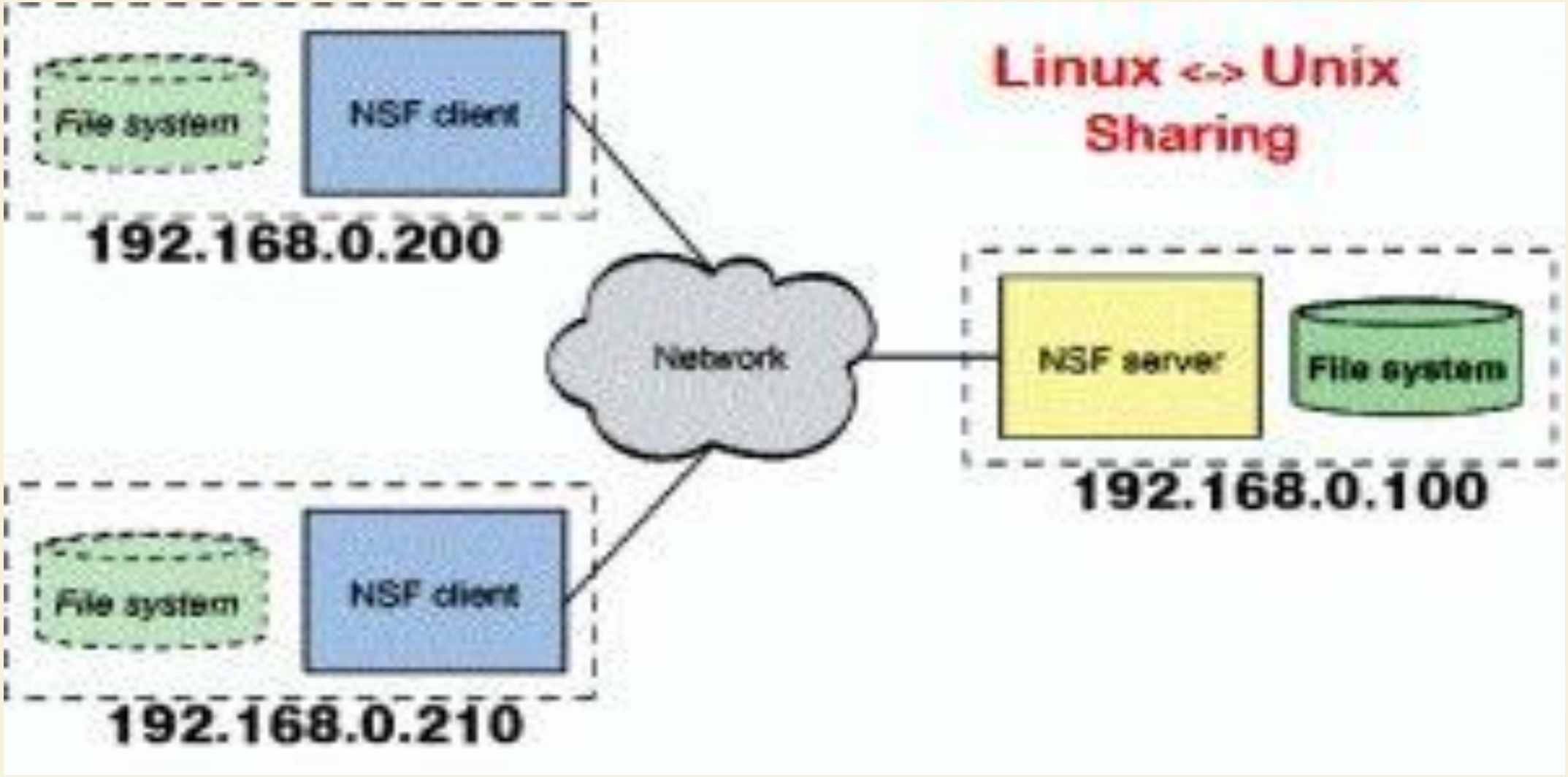
The client machine requests access to exported data, typically by issuing a mount command.

If all goes well, users on the client machine can then view and interact with mounted file systems on the server within the parameters permitted.

To mount is to make a group of files in a file system structure accessible to a user or user group. In some usages, it means to make a device physically accessible. For instance, in data storage, to ***mount*** *is to place a data medium (such as a tape cartridge) on a drive in a position to operate*

mount displays a list of file systems that are currently mounted on your computer

PROTOCOLS CONT'D



PROTOCOLS CONT'D

iv) SMTP

Simple Mail Transfer Protocol (SMTP) is the standard for e-mail transmissions across the Internet developed during 1970's.
SMTP is a relatively simple, text-based protocol, in which one or more recipients of a message are specified (and in most cases verified to exist) and then the message text is transferred.
It is a Client/Server protocol, whereby a client transmits an e-mail message to a server.
Either an end-user's e-mail client, a.k.a. MUA (Mail User Agent), or a relaying server's MTA (Mail Transfer Agents) can act as an SMTP client.
An email client knows the outgoing mail SMTP server from its configuration.

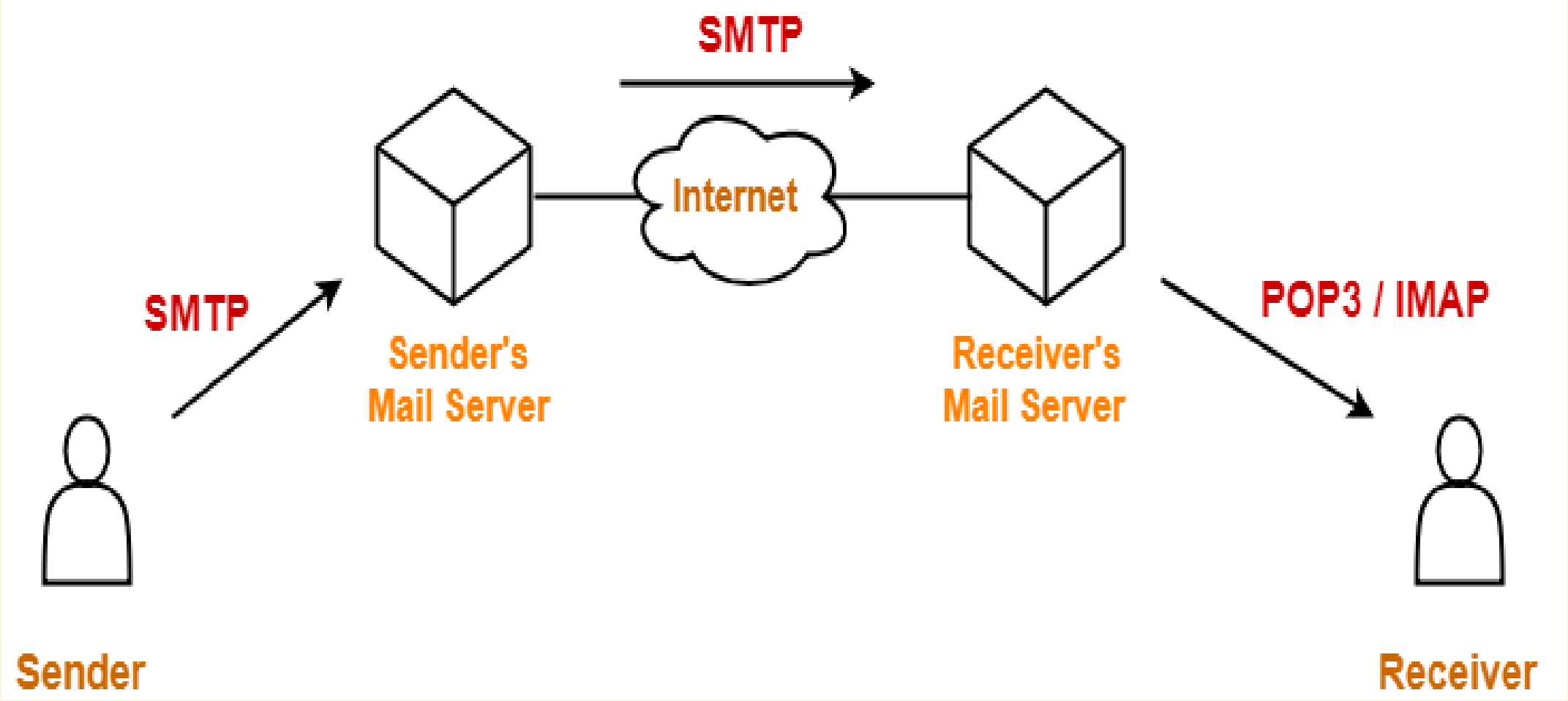
PROTOCOLS CONT'D

A relaying server typically determines which SMTP server to connect to by looking up the MX (Mail eXchange) DNS record for each recipient's domain name (the part of the e-mail address to the right of the at (@) sign).
Conformant MTAs (not all) fall back to a simple A record in the case of no MX.
Some current mail transfer agents will also use SRV records, a more general form of MX, though these are not widely adopted.
(Relaying servers can also be configured to use a smart host.

PROTOCOLS CONT'D

SRV (Service) records are custom DNS records used to establish connections between a service and a hostname
SMTP is a "push" protocol that does not allow one to "pull" messages from a remote server on demand.
To do this a mail client must use POP3 or IMAP.
Another SMTP server can trigger a delivery in SMTP using ETRN.
An e-mail client requires the name or the IP address of an SMTP server as part of its configuration.
The server will deliver messages on behalf of the user

PROTOCOLS CONT'D



PROTOCOLS CONT'D

This setting allows for various policies and network designs. End users connected to the Internet can use the services of an e-mail provider that is not necessarily the same as their connection provider.

Network topology, or the location of a client within a network or outside of a network, is no longer a limiting factor for e-mail submission or delivery.

Modern SMTP servers typically use a client's credentials (authentication) rather than a client's location (IP address), to determine whether it is eligible to relay e-mail. One of the limitations of the original SMTP is that it has no facility for authentication of senders.

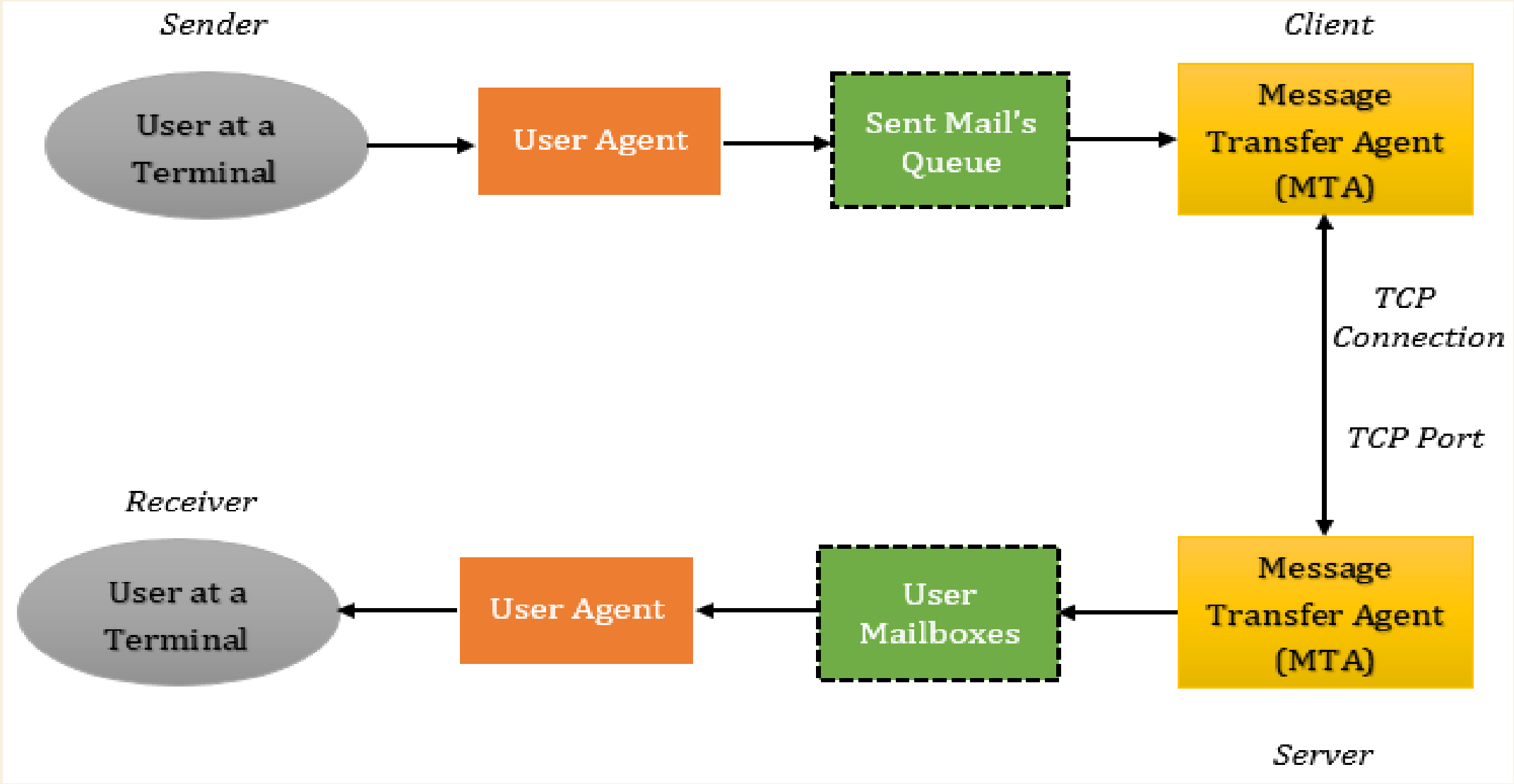
PROTOCOLS CONT'D

Therefore, the SMTP-AUTH extension was defined. However, the impracticalities of widespread SMTP-AUTH implementation and management means that E-mail spamming is not and cannot be addressed by it.

SMTP Authentication, often abbreviated SMTP AUTH, is an extension of the Simple Mail Transfer Protocol (SMTP) whereby a client may log in using any authentication mechanism supported by the server.

It is mainly used by submission servers, where authentication is mandatory.

PROTOCOLS CONT'D



INTERPOSES COMMUNICATION

The communication between two processes take place via buffer.

The alternative way of communication is the process of the inter-process communication.

The simple mechanism of this is synchronizing their action and without sharing the same address space.

This play an important role in the distributed processing environment.

While signals, pipes and names pipes are ways by which processes can communicate.

The more redefined method of inter process communication are message queues, semaphores and shared memory

INTERPOSES COMMUNICATION

Client/Server communication involves two components, namely a client and a server.
They are usually multiple clients in communication with a single server.
The clients send requests to the server and the server responds to the client requests.
There are three main methods to client/server communication:

=> Socket
=> Remote procedure call
=> Pipes

INTERPOSES COMMUNICATION CONT'D

Sockets

Sockets facilitate communication between two processes on the same machine or different machines.

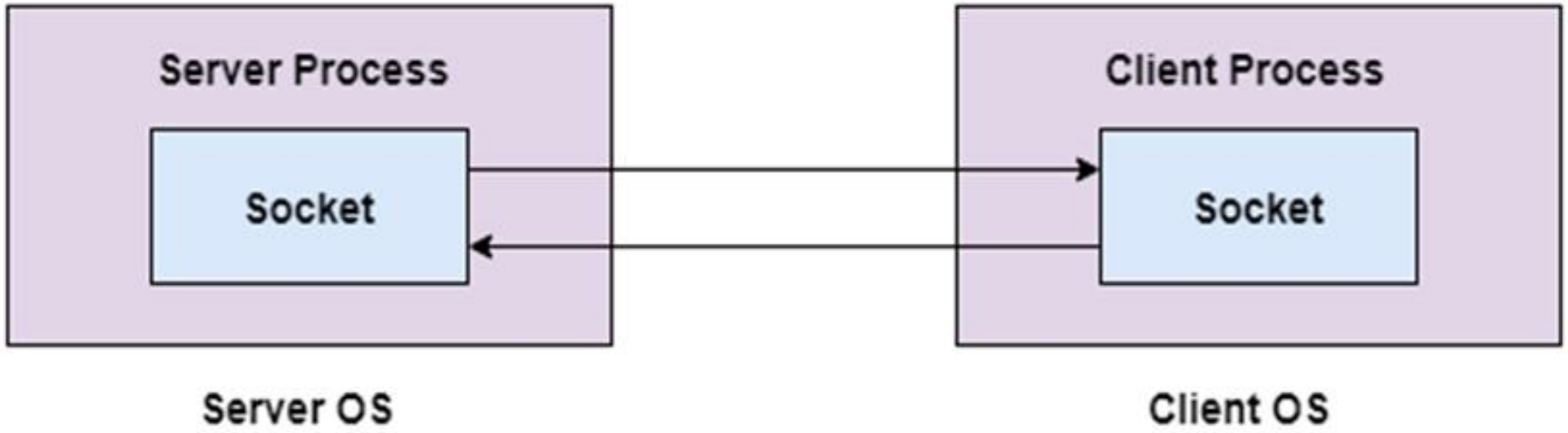
They are used in a client/server framework and consist of the IP address and port number.

Many application protocols use sockets for data connection and data transfer between a client and a server.

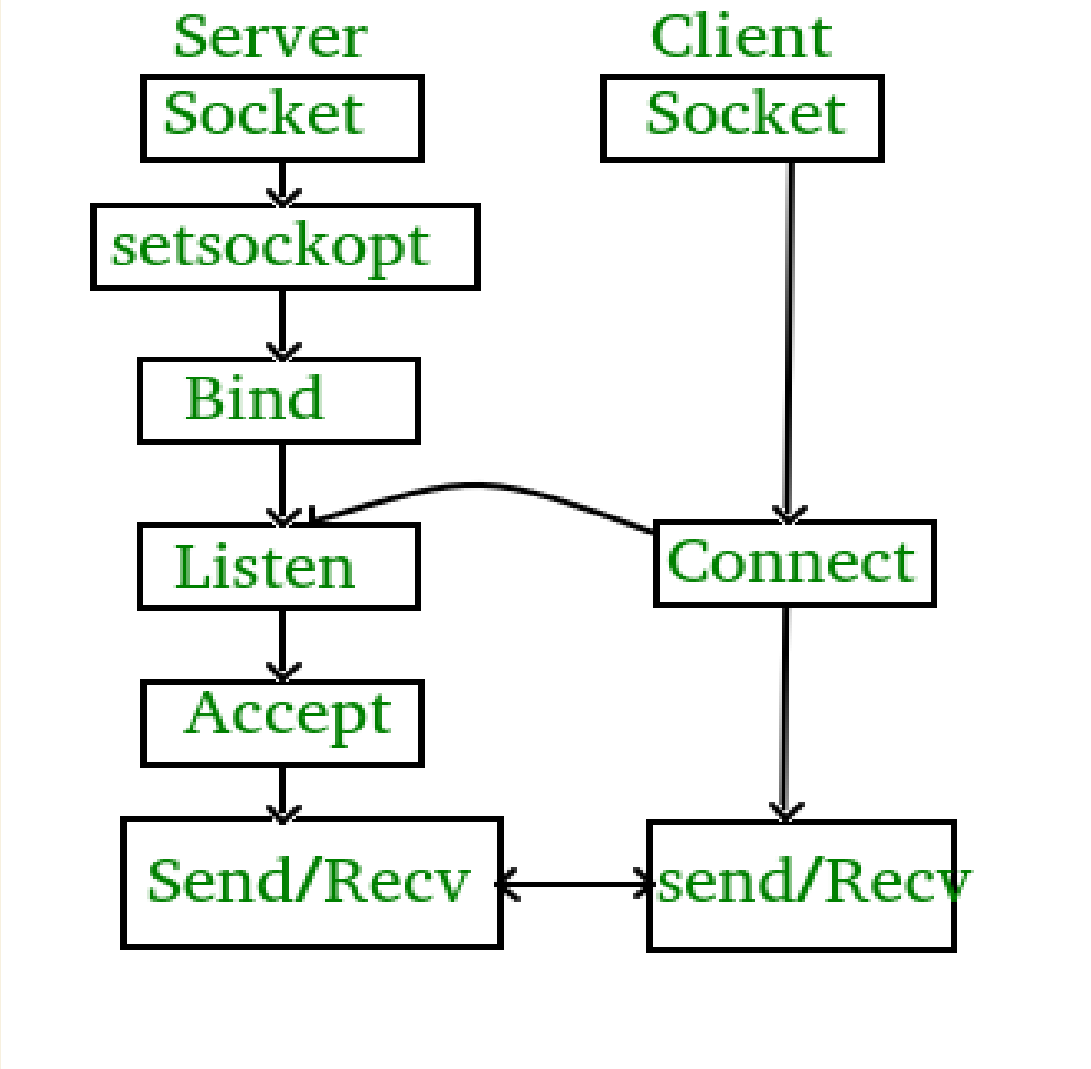
Socket communication is quite low-level as sockets only transfer an unstructured byte stream across processes. The structure on the byte stream is imposed by the client and server applications

INTERPOSES COMMUNICATION CONT'D

A diagram that illustrates sockets is as follows –



INTERPOSES COMMUNICATION CONT'D



Socket programming is a way of connecting two nodes on a network to communicate with each other.

One socket(node) listens on a particular port at an IP, while other socket reaches out to the other to form a connection.

Server forms the listener socket while client reaches out to the server.

INTERPOSES COMMUNICATION CONT'D

Remote Procedure Calls

These are inter-process communication techniques that are used for client-server based applications.

A remote procedure call is also known as a subroutine call or a function call.

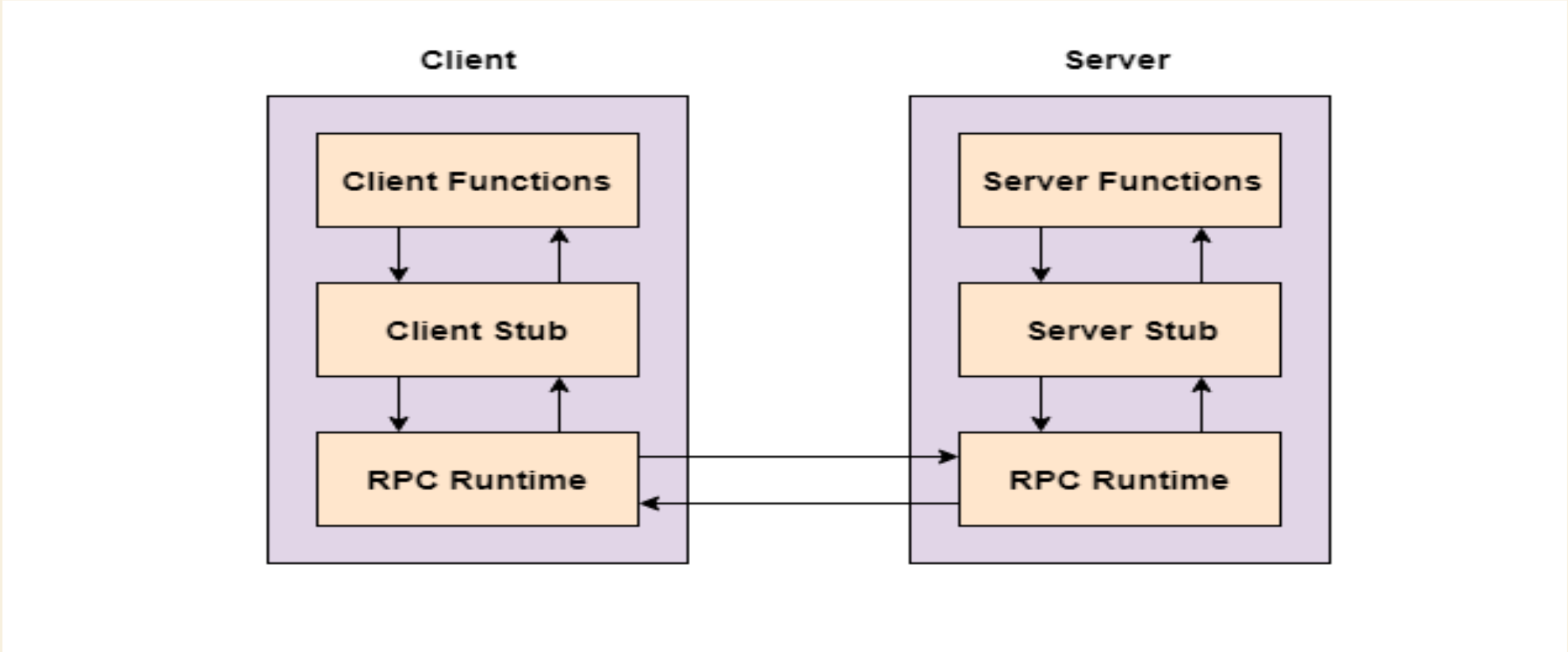
A client has a request that the RPC translates and sends to the server.

This request may be a procedure or a function call to a remote server.

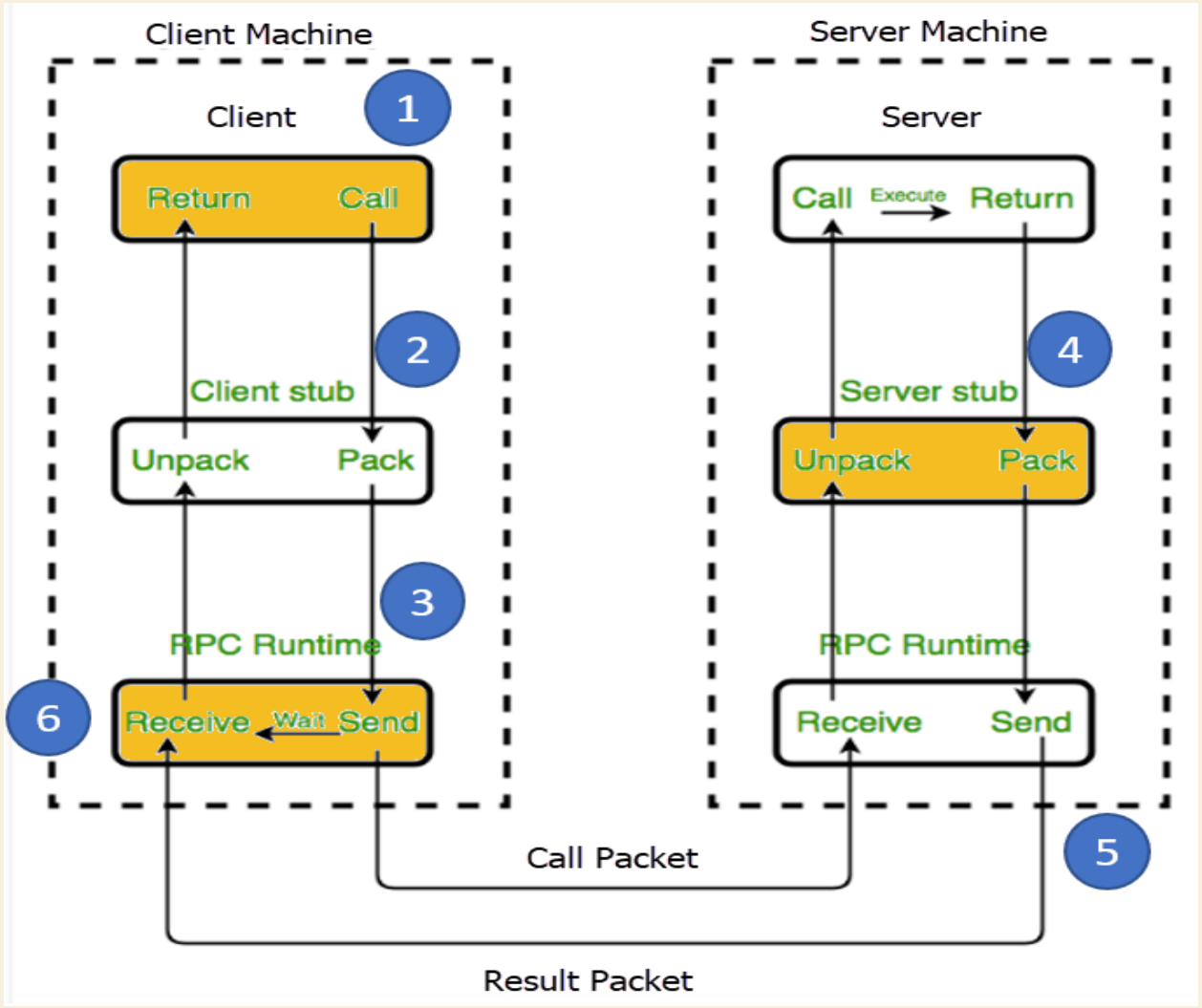
When the server receives the request, it sends the required response back to the client.

INTERPOSES COMMUNICATION CONT'D

A diagram that illustrates remote procedure calls is given as follows –



INTERPOSES COMMUNICATION CONT'D



RPC mechanisms are used when a computer program causes a procedure or subroutine to execute in a different address space, which is coded as a normal procedure call without the programmer specifically coding the details for the remote interaction

INTERPOSES COMMUNICATION CONT'D

Pipes

These are inter-process communication methods that contain two end points.

Data is entered from one end of the pipe by a process and consumed from the other end by the other process.

The two different types of pipes are ordinary pipes and named pipes.

- Ordinary pipes only allow one-way communication.
- For two-way communication, two pipes are required.

Ordinary pipes have a parent child relationship between the processes as the pipes can only be accessed by processes that created or inherited them.

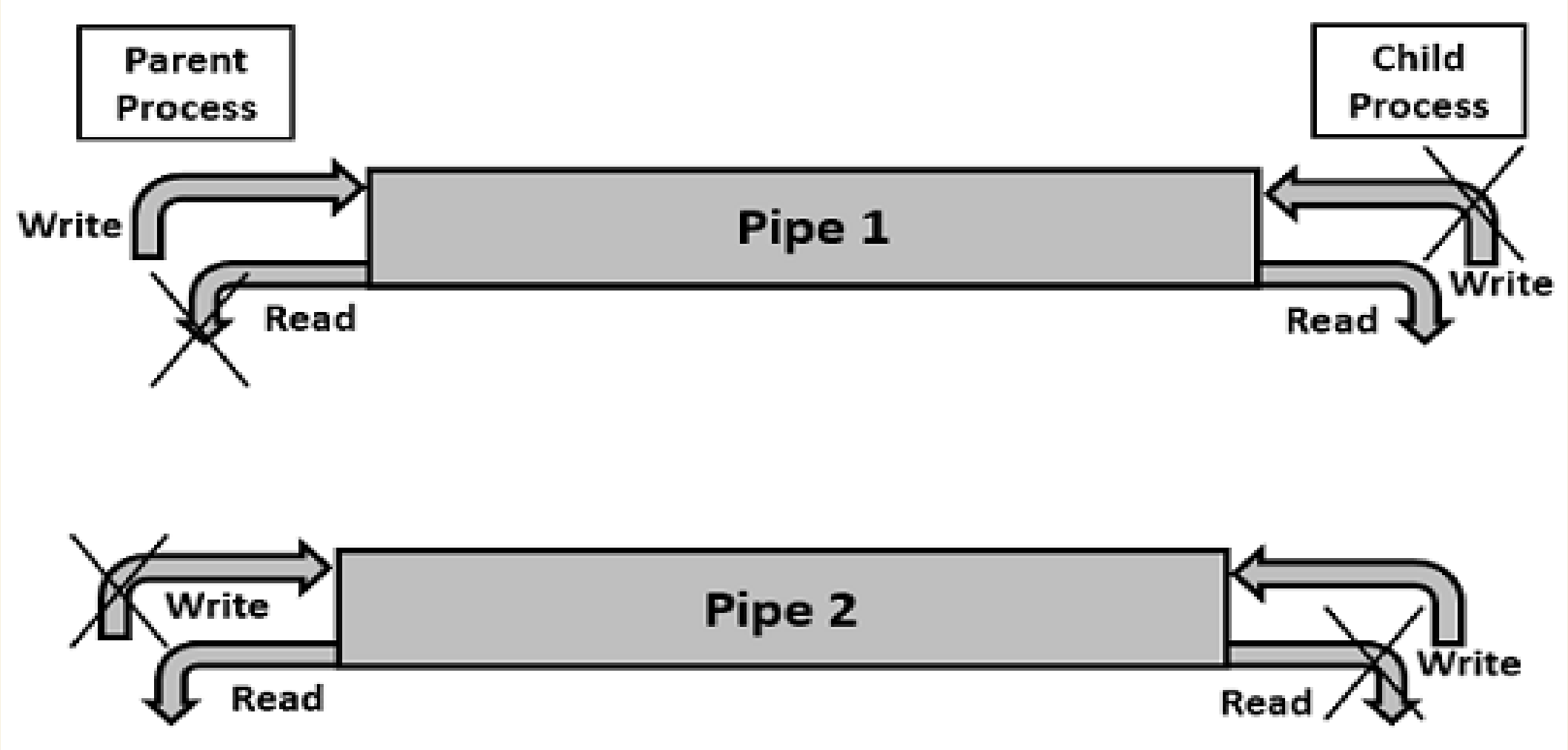
INTERPOSES COMMUNICATION CONT'D

Named pipes are more powerful than ordinary pipes and allow two-way communication.

These pipes exist even after the processes using them have terminated.

They need to be explicitly deleted when not required anymore.

INTERPOSES COMMUNICATION CONT'D



INTERPOSES COMMUNICATION CONT'D

The more redefined method of inter process communication are message queues, semaphores and shared memory.

There are four types of mechanisms, involved for such communications: -

(i)Message passing.

(ii)Direct communication.

(iii)Indirect communication.

(iv)Remote procedures call.

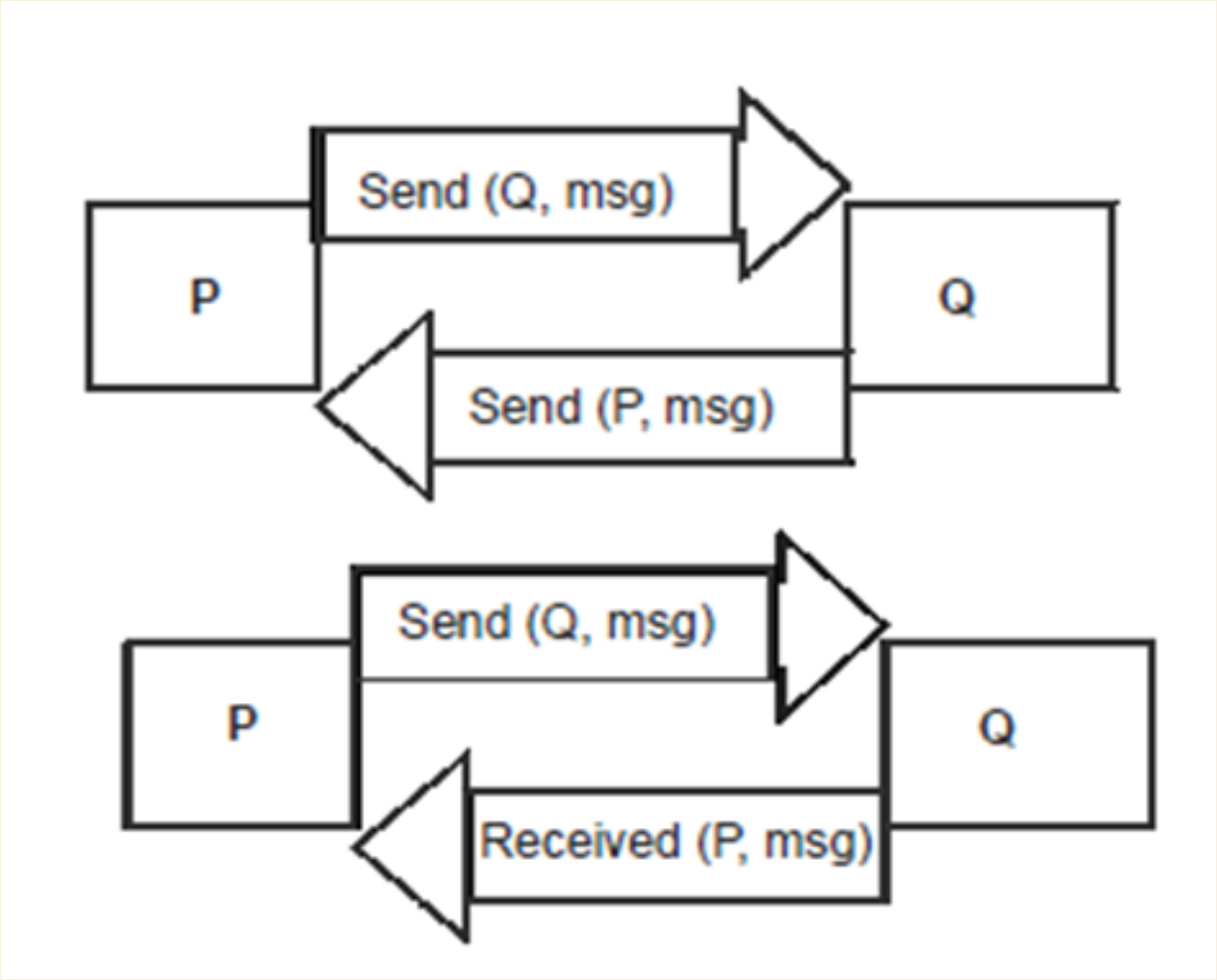
INTERPOSES COMMUNICATION CONT'D

i) Message passing:

This mechanism allows process to communicate without restoring the shared data, for example in micro kernel, message is passed to communicate in which services acts as an ordinary user where these services act outside the kernel. At least, there are two processes involved in an IPC.

The following figure illustrates message passing mechanism.

INTERPOSES COMMUNICATION CONT'D



=>Sending process for sending the message.
=>Receiving process for receiving the message.
Messages sent by the processes are of two types, fixed and variable.
For the communication to be taking place, a link is to be set in between the two processes

INTERPOSES COMMUNICATION CONT'D

ii) Direct communication

In this mechanism of communication processes have to specify the name of sender and recipient process name. This type of communication has the following features:

A link is established in between the sender and receiver along with full known information of their names and addresses.

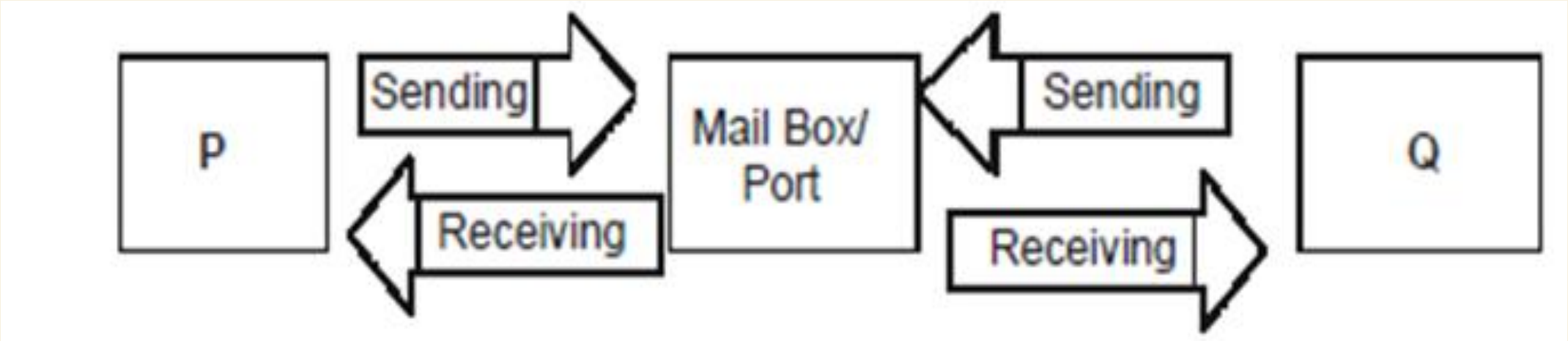
One link must be established in between the processes.

There is symmetry in between the communication of the processes

INTERPOSES COMMUNICATION CONT'D

iii) Indirect communication

In indirect communication, messages are sending to the mail box and then they are retrieved from mailbox



INTERPOSES COMMUNICATION CONT'D

The role of the mailbox is quite similar to the role of the postman.

The indirect communication can also communicate with other processes via one or more mailbox.

The following features are associated with indirect communication:

=>A link is established between a pair of process, if they share a mailbox.

=> A link is established between more than one processes.

=>Different number of links can be established in between the two communicating processes.

INTERPOSES COMMUNICATION CONT'D

Communication between the processes takes place by executing calls to the send and receive primitive.
Now there is several different ways to implement these primitives, they can be "blocking" and "non-blocking".
The different possible combinations are:

- Blocking send: Sending the process is blocked until the message is received.
- Non-blocking send: In it process sends the message and then it resumes the operation.
- Blocking receive: Receiver is blocked until the message is available.
- Non-blocking receive: The receiver receives either a valid message or a null.

INTERPOSES COMMUNICATION CONT'D

Remote procedures call

RPC is a powerful technique for constructing distributed, client-server based applications.

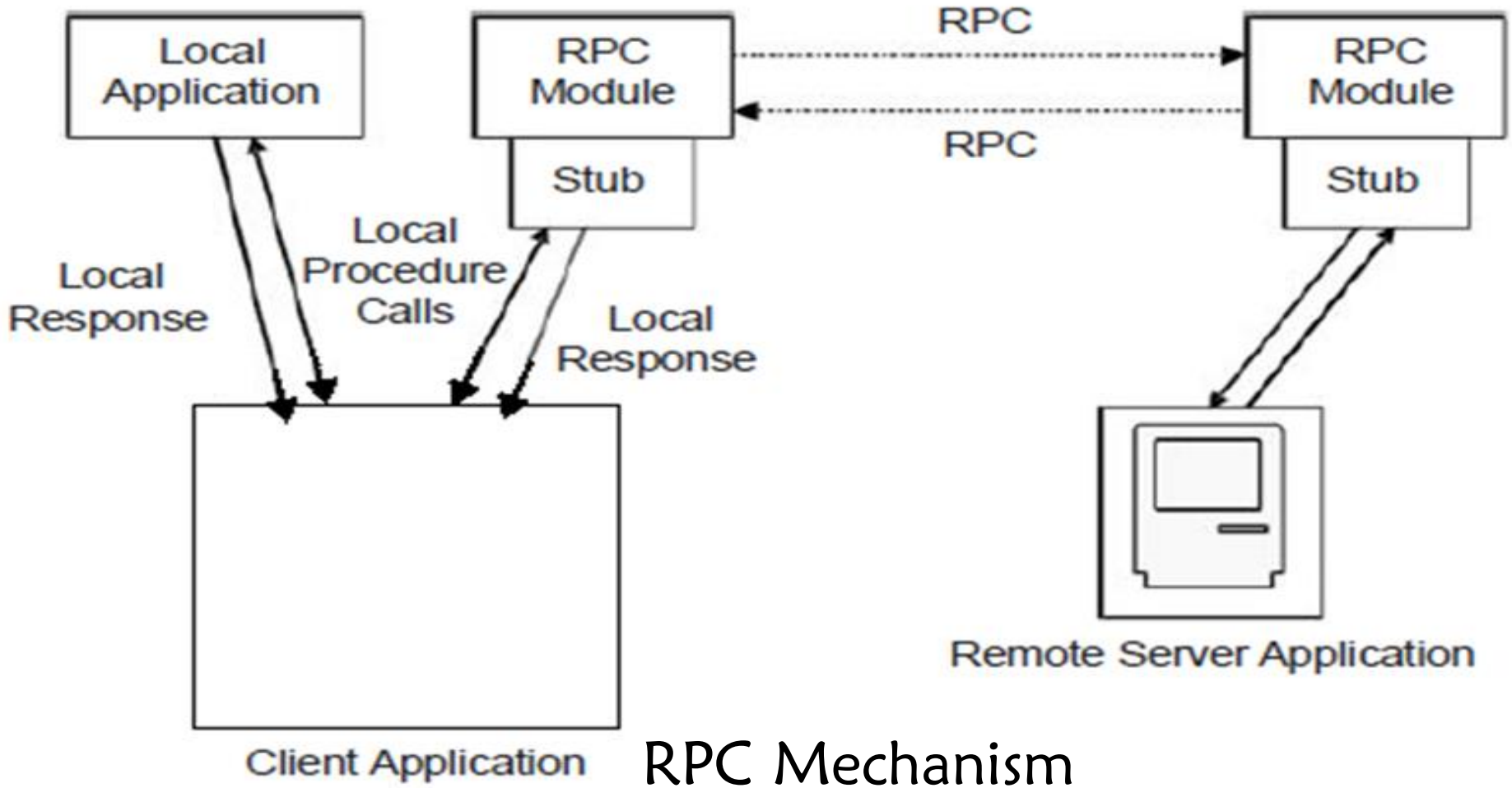
The essence of the technology is to allow programs on different machines to interact using simple procedure call or return semantics, just as if the two programs were on the same machine.

It is based on extending the notion of conventional or local procedure calling, so that the called procedure need not exist in the same address space as the calling procedure.

The two processes may be on the same system, or they may be on different systems with a network connecting them.

That is, the procedure call is used for access to remote services

INTERPOSES COMMUNICATION CONT'D



RPC Mechanism

INTERPOSES COMMUNICATION CONT'D

A remote procedure is uniquely identified by the triple: (program number, version number, procedure number),
=> the program number identifies a group of related remote procedures, each of which has a unique procedure number. A program may consist of one or more versions.
=> Each version consists of a collection of procedures which are available to be called remotely. Version numbers enable multiple versions of an RPC protocol to be available simultaneously. Each version contains a number of procedures that can be called remotely.
=> Each procedure has a procedure number. Procedure may or may not be transparent to the user that the intention is to invoke a remote procedure on the same machine.

INTERPOSES COMMUNICATION CONT'D

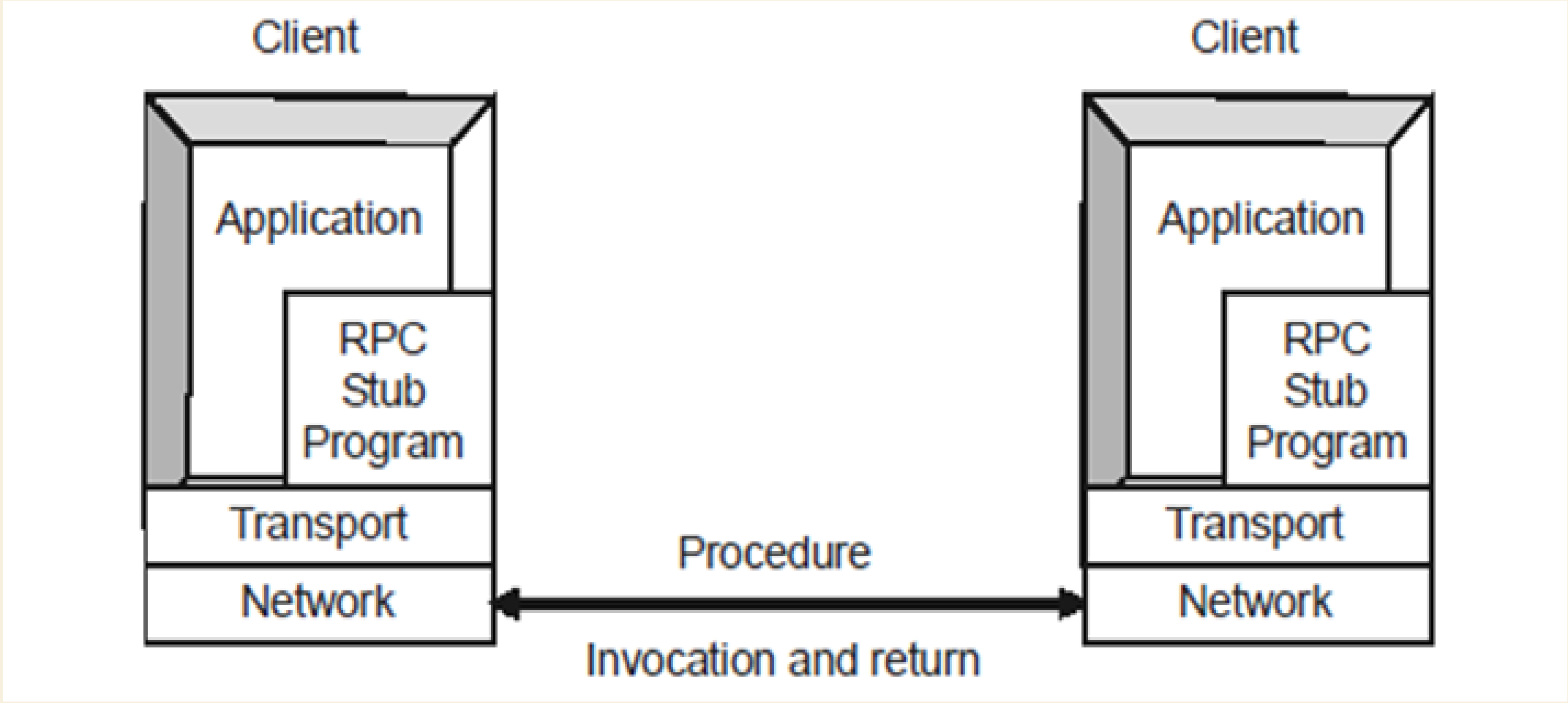
RPCs, are APIs, layered on top of a network IPC mechanism, allows users to communicate users directly with each other's. They allow individual processing components of an application to run other nodes in the network.

Distributed file systems, system management, security and application programming depend on the capabilities of the underlying RPC mechanisms.

Server access control and use of a directory service are common needs that can be met with RPCs.

RPCs also manage the network interface and handle security and directory services.

INTERPOSES COMMUNICATION CONT'D



General Architecture

INTERPOSES COMMUNICATION CONT'D

Here, the structure of RPC allows a client to invoke a procedure on the remote host locally, which is done with the help of "stub" which is provided by the client.

Thus, when the client invokes the remote procedure RPC calls the appropriate stub, passes the parameters to it, which are then provided, to remote procedure.

This stub locates the port on the server and marshalling involves packaging the parameter into a form, which may be transmitted over network.

The stub then transmits a message to server using message passing.

Now the message sent by the host is received at the client side with the help of similar type of stub

INTERPOSES COMMUNICATION CONT'D

Limitation of RPC:

There are number of limitations associated with RPC given below.

1. RPC requires synchronous connections. If an application uses an RPC to link to a server that is busy that time then application will have to wait for the data rather than switching to other task.

2. Local procedure call fails under the circumstances where RPC can be duplicated under and executed more than one, which is due to unreliable communication.

INTERPOSES COMMUNICATION CONT'D

3. The communication in between the client and server is done with help of the standard procedure calls; therefore, some binding must take place during the link load and execution, such that the process is replaced by the address. The RPC binds the same thing to the client and server.

A general problem that exists is that there is no shared memory in between them so how they can come to know about the address of the other system.

INTERPOSES COMMUNICATION CONT'D

4. The binding information may be predetermined in the form of the port address, at the compile time, a RPC call, that has a fix port number is associated with it. Once a program is compiled, it then cannot change its port number.
5. Binding can be done dynamically by rendezvous mechanism. Typically, an operating system provides rendezvous demon requesting the port address of RPC, it needed to execute. The port address is then returned and the RPC call may be sent to the port until the process terminates.

MIDDLEWARE

Middleware in the context of distributed applications is software that provides services beyond those provided by the operating system to enable the various components of a distributed system to communicate and manage data.

Middleware supports and simplifies complex distributed applications

Middleware is software which lies between an operating system and the applications running on it.

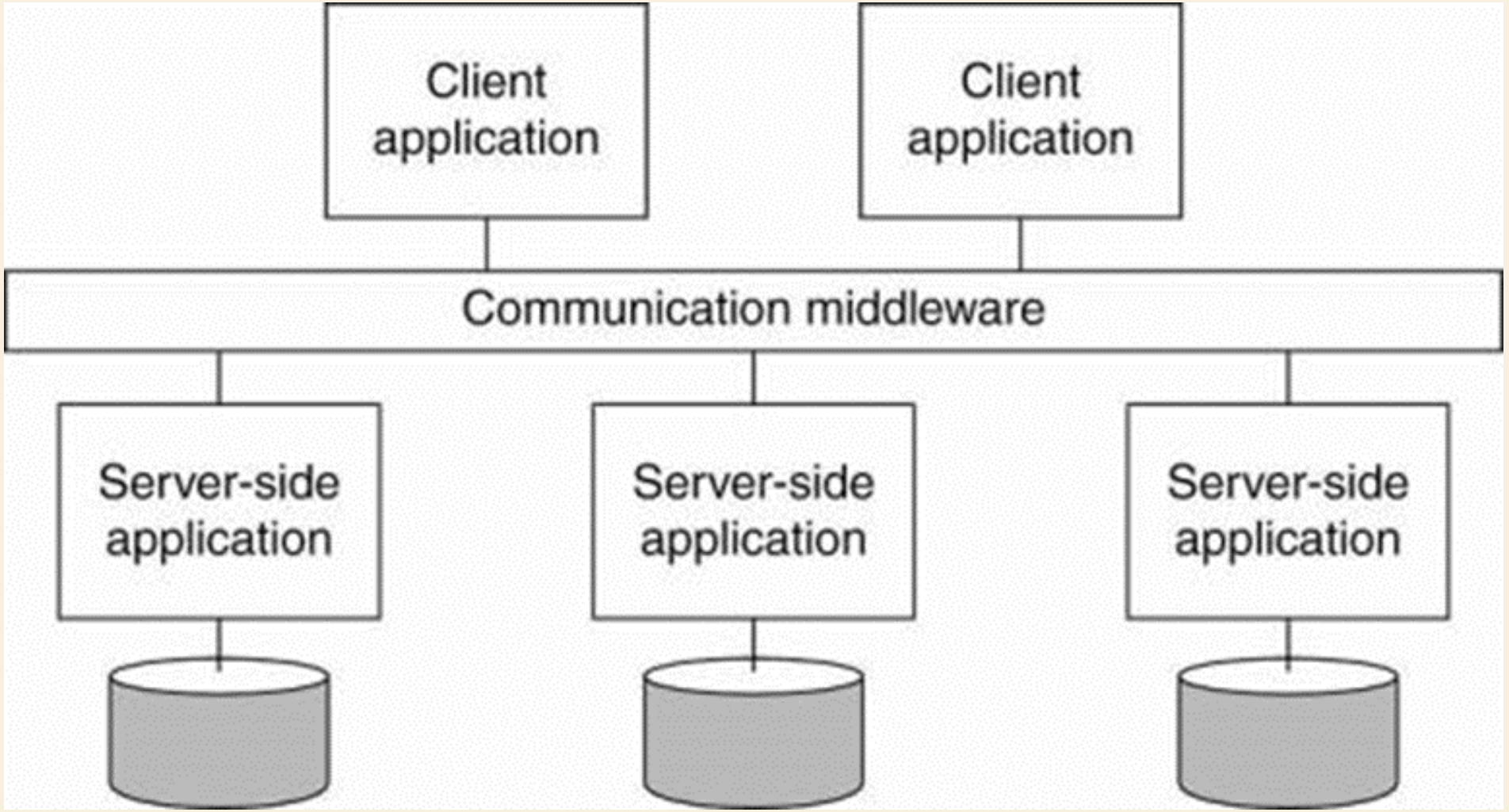
MIDDLEWARE CONT'D

Essentially functioning as hidden translation layer, middleware enables communication and data management for distributed applications.

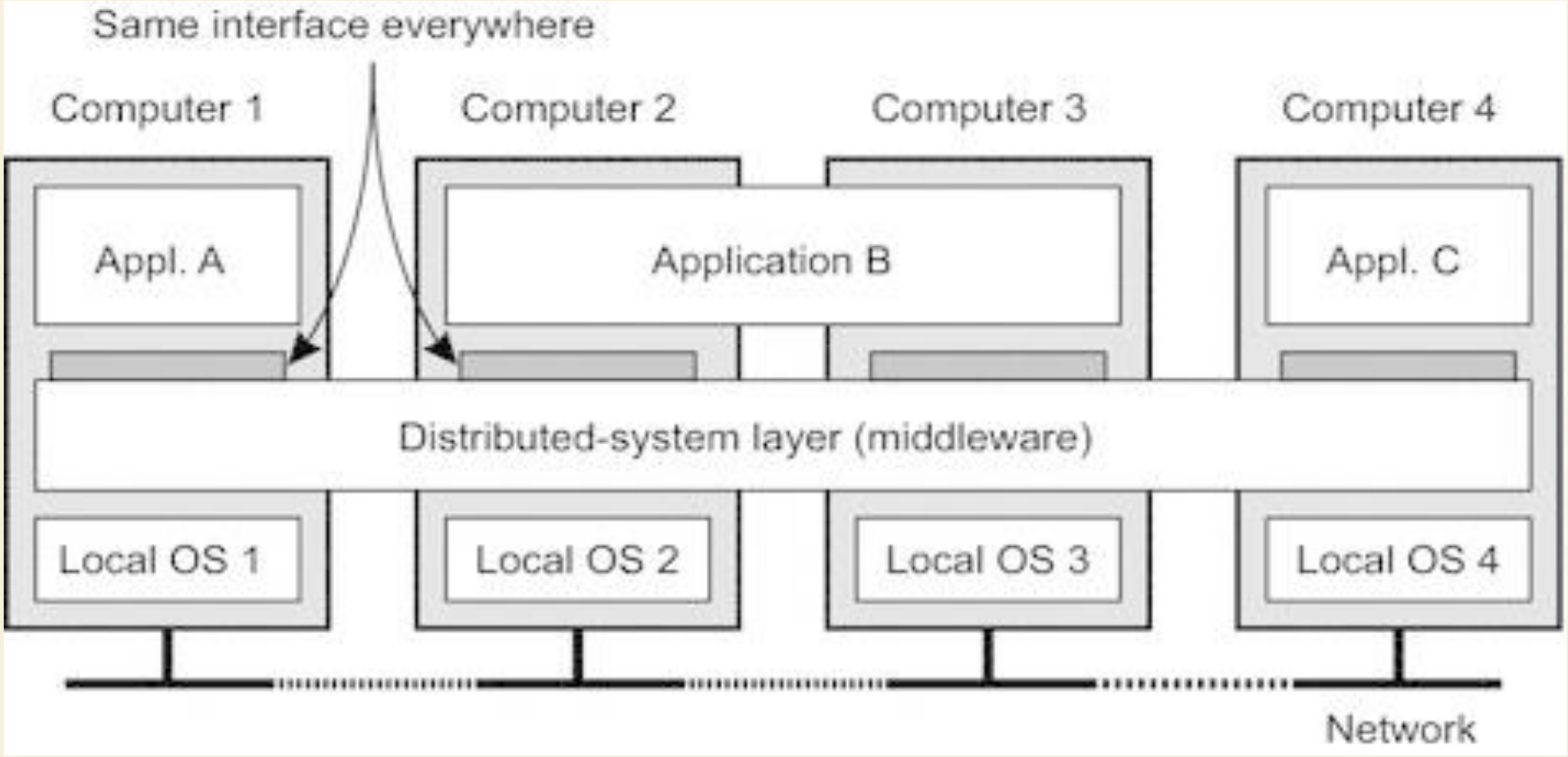
It is sometimes called plumbing, as it connects two applications together so data and databases can be easily passed between the "pipe."

Using middleware allows users to perform such requests as submitting forms on a web browser or allowing the web server to return dynamic web pages based on a user's profile.

MIDDLEWARE CONT'D



MIDDLEWARE CONT'D



MIDDLEWARE CONT'D

Common middleware examples include database middleware, application server middleware, message-oriented middleware, web middleware and transaction-processing monitors.

Each program typically provides messaging services so that different applications can communicate using messaging frameworks like simple object access protocol (SOAP), web services, representational state transfer (REST) and JavaScript object notation (JSON).

MIDDLEWARE CONT'D

While all middleware performs communication functions, the type a company chooses to use will depend on what service is being used and what type of information needs to be communicated.

This can include security authentication, transaction management, message queues, applications servers, web servers and directories.

Middleware can also be used for distributed processing with actions occurring in real time rather than sending data back and forth

MIDDLEWARE CONT'D

i) Message Oriented Middleware (MOM)

This is a large category and includes communication via message exchange.

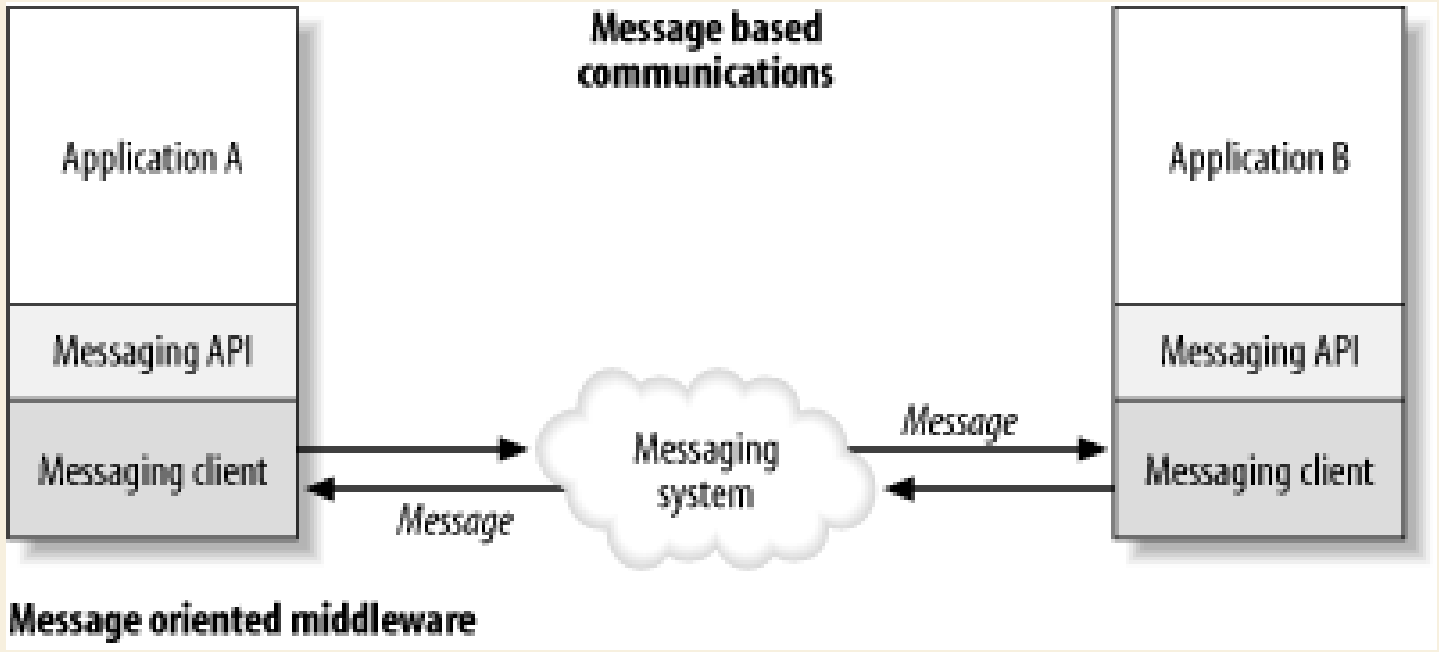
It represents asynchronous interactions between systems.

It reduces complexity of developing applications that span multiple operating systems and network protocols by insulating the application developer from details of various operating system and network interfaces.

APIs that extend across diverse platforms and networks are typically provided by the MOM.

MIDDLEWARE CONT'D

Message Oriented Middleware is a concept that involves the passing of data between applications using a communication channel that carries self-contained units of information (messages).



MIDDLEWARE CONT'D

MOM is software that resides in both portions of client/server architecture and typically supports asynchronous calls between the client and server applications.

Message queues provide temporary storage when the destination program is busy or not connected.

MOM reduces the involvement of application developers with the complexity of the master-slave nature of the client/server mechanism. E.g. Sun's JMS.

MIDDLEWARE CONT'D

ii) Object Request Broker (ORB)

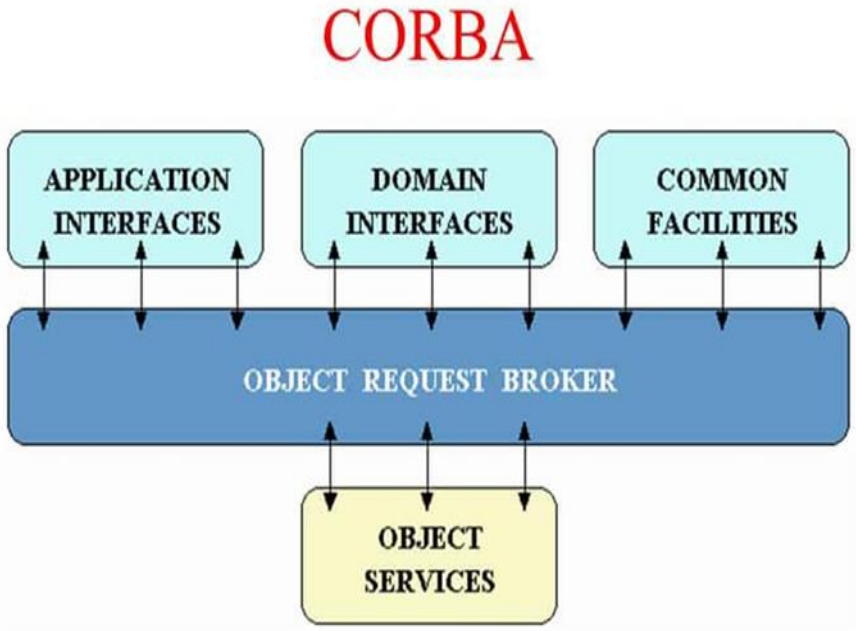
In distributed computing, an object request broker (ORB) is a piece of middleware software that allows programmers to make program calls from one computer to another via a network.

ORB's handle the transformation of in-process data structures to and from the byte sequence, which is transmitted over the network.

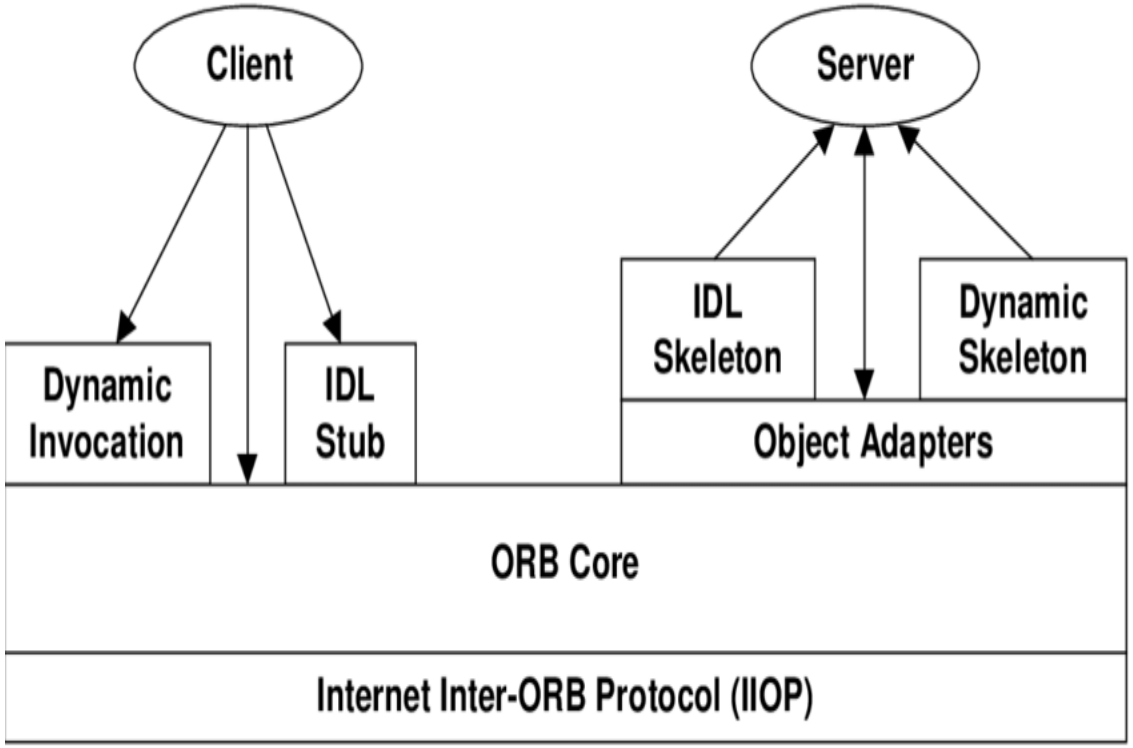
This is called marshaling or serialization.

Some ORB's, such as CORBA-compliant systems, use an Interface Description Language (IDL) to describe the data which is to be transmitted on remote calls. E.g. CORBA

MIDDLEWARE CONT'D



OMG Reference Model architecture



MIDDLEWARE CONT'D

iii) RPC Middleware.

This type of middleware provides for calling procedures on remote systems, so called as Remote Procedure Call.

Unlike message oriented middleware, RPC middleware represents synchronous interactions between systems and is commonly used within an application.

Thus, the programmer would write essentially the same code whether the subroutine is local to the executing program, or remote.

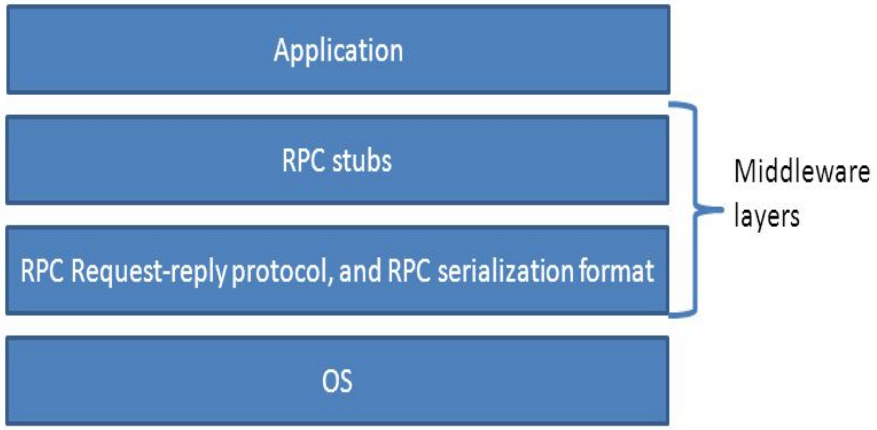
When the software in question is written using object-oriented principles, RPC may be referred to as remote invocation or remote method invocation.

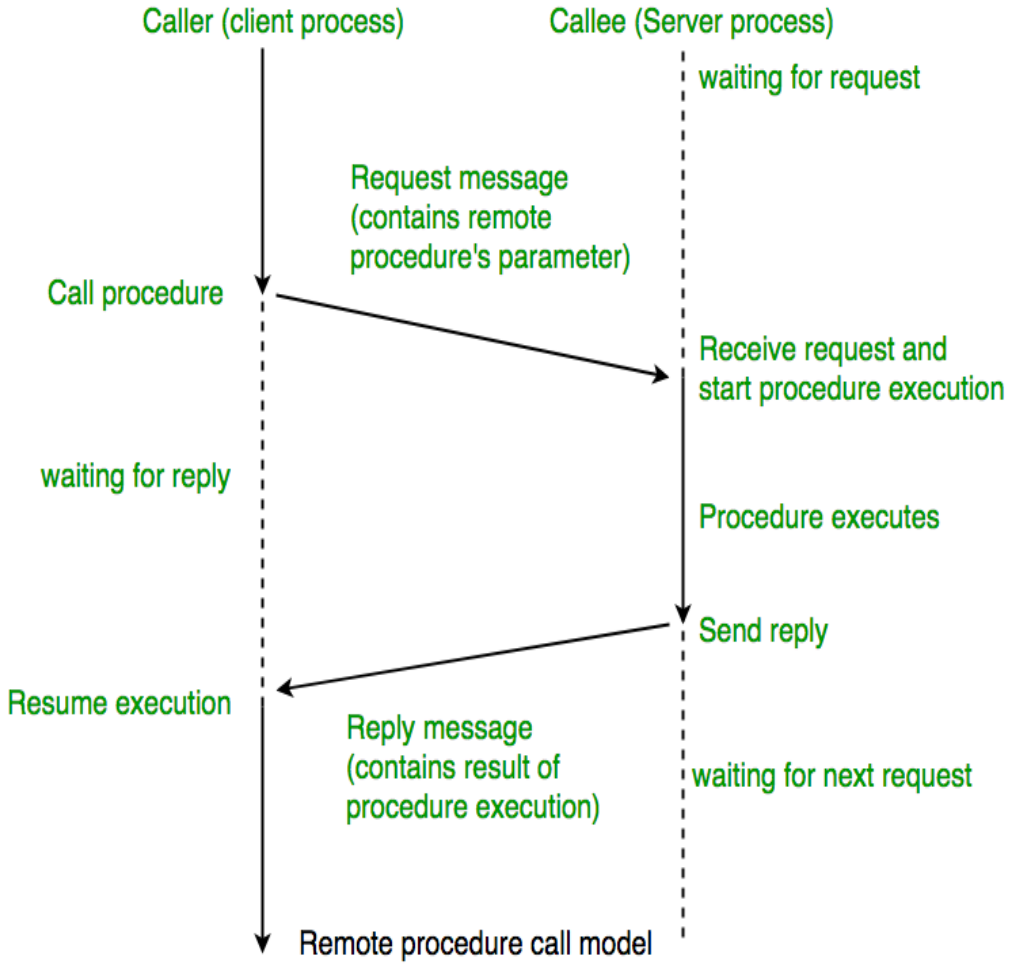
Client makes calls to procedures running on remote systems, which can be asynchronous or synchronous. E.g. DCE RPC.

MIDDLEWARE CONT'D

RPC as middleware

Middleware: software between OS and application





MIDDLEWARE CONT'D

iv) Database Middleware.

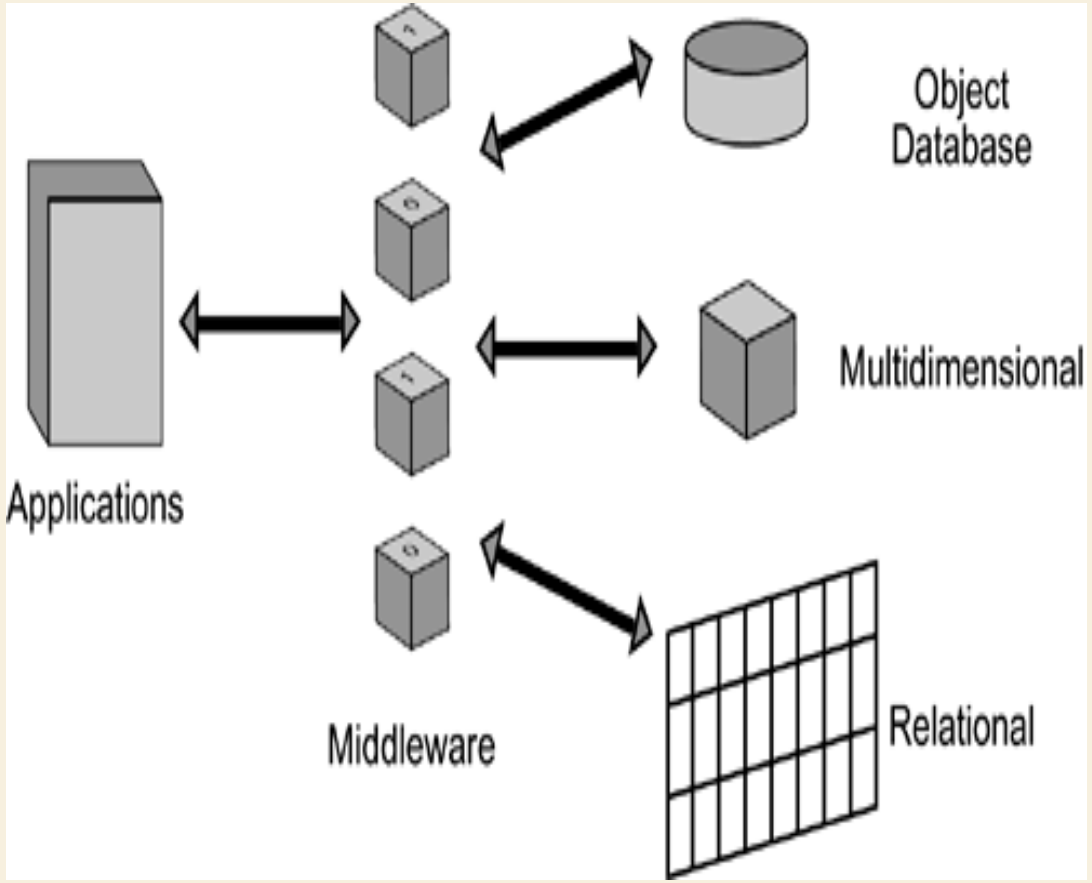
Database middleware allows direct access to data structures and provides interaction directly with databases.

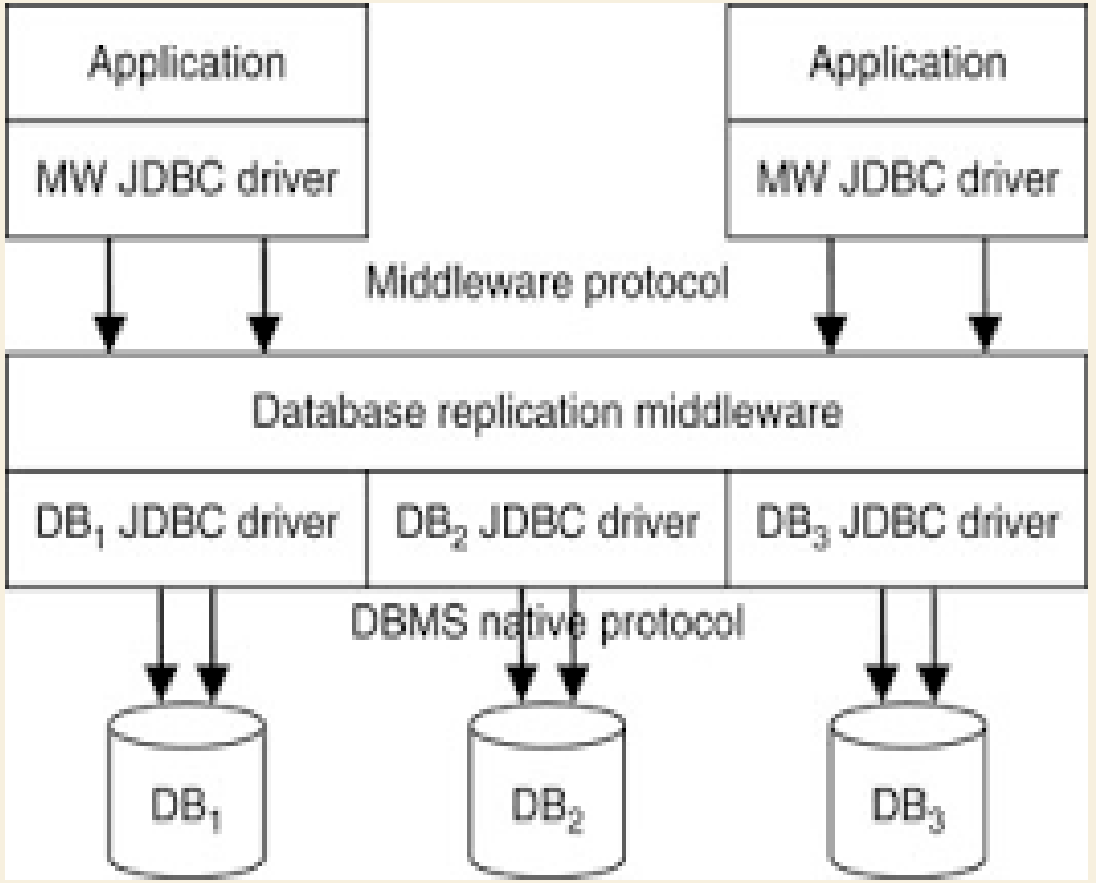
There are database gateways and a variety of connectivity options.

Extract, Transform, and Load (ETL) packages are included in this category.

E.g. CRAVE is a web-accessible JAVA application that accesses an underlying MySQL database of ontologies via a JAVA persistent middleware layer Chameleon).

MIDDLEWARE CONT'D





MIDDLEWARE CONT'D

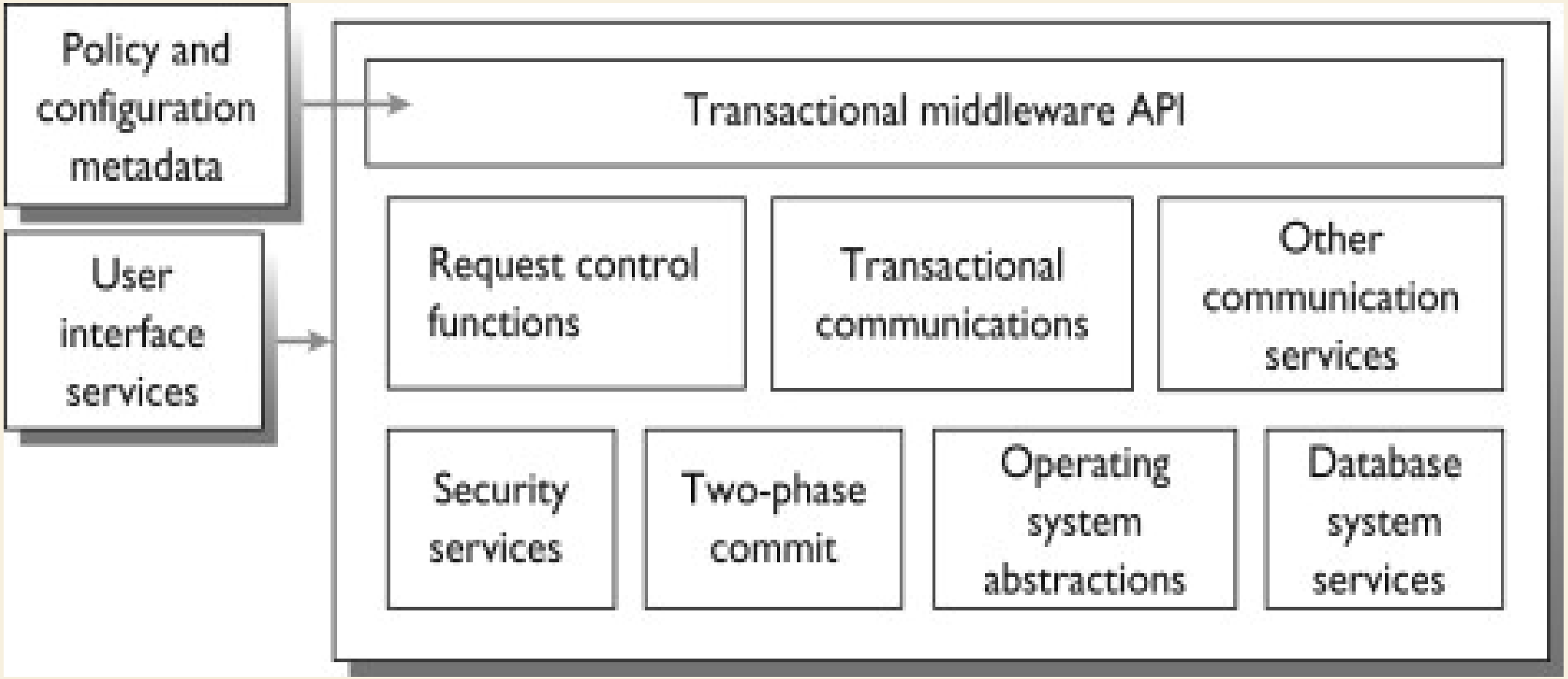
v) Transaction Middleware.

This category as used in the Middleware Resource Center includes traditional transaction processing monitors (TPM) and web application servers. e.g. IBM's CICS. F.

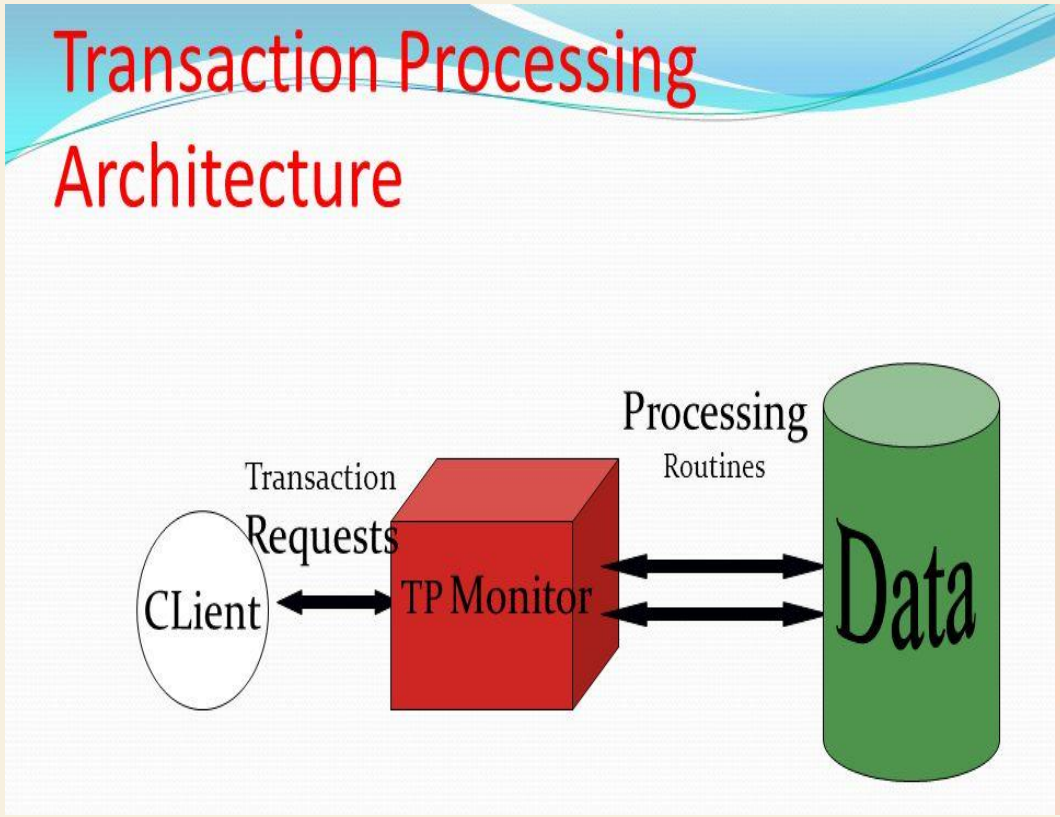
Portals :

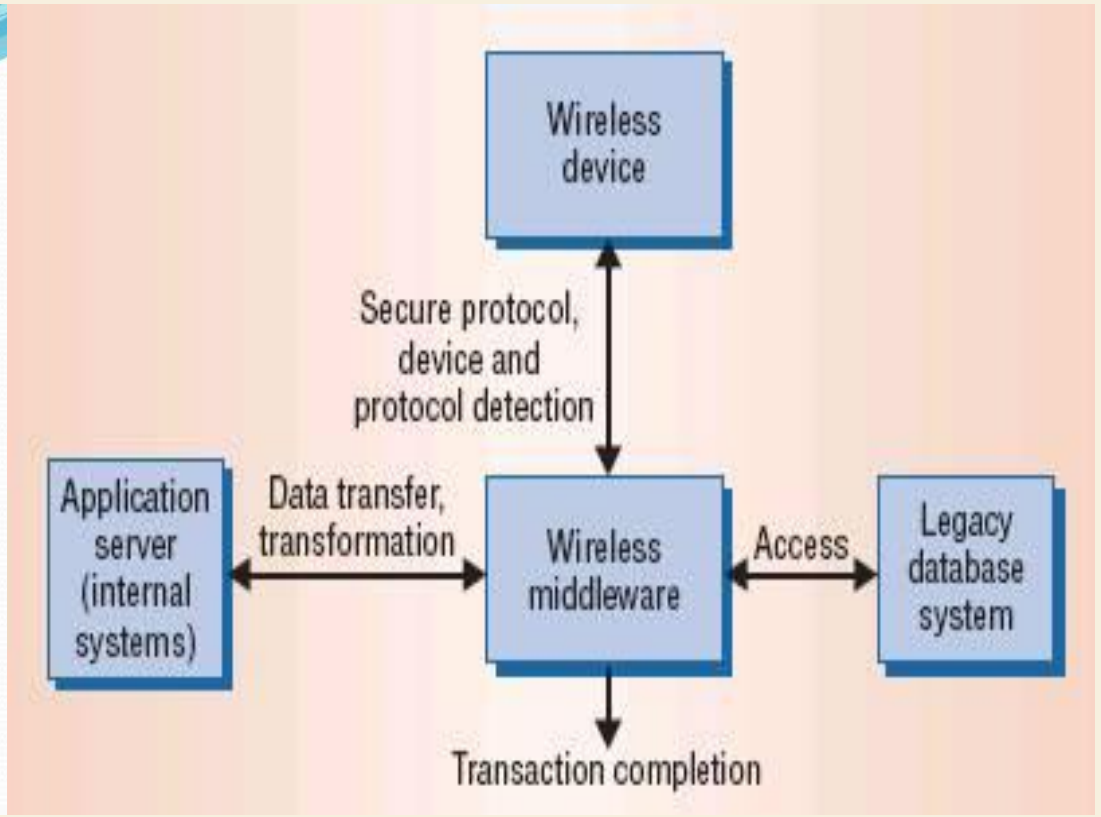
Enterprise portal servers are included as middleware largely because they facilitate "front end" integration. They allow interaction between the user's desktop and back end systems and services. e.g Web Logic

MIDDLEWARE CONT'D



MIDDLEWARE CONT'D





Wireless middleware is a software layer which helps coordination between programs, OSs, hardware platforms and communication protocols

SUMMARY

We have discussed the following:

i) User interfaces
   - Application programming interface (API)

ii) Protocols
   - TCP/IP, SNMP, NFS,SMTP

iii) Inter-process Communication
   - message passing, direct communication, indirect communication and remote procedure call

iv) Middleware
   - Message oriented middleware, object request broker middleware, RPC middleware, database middleware and transactional

The End !!!

?