➢ Middleware: Models of Middleware

The first middle ware model started with the distributed file system, where the files were stored and distributed over the network.

Models of middle ware are as follows:

Remote Procedure call:

It is one of the successful middle ware models used in modern distributed applications for communication.

It uses a local call to call a procedure residing on the remote machine to get the result.

Hidden communication is done between client and server.

For eg If a user wants to get the sum of two numbers stored on a remote server by using local method call, the user calls method with parameters, in turn server receives a RPC call from the client and returns the appropriate result to the client.

Therefore, though the method was executed remotely it appears like a local to the called machine.

This is a synchronous technology where both the client and server should be present during the communication.

Message oriented Middleware(MOM):

It is another model used to send and receive the communication messages between clients and servers. It uses data structures like queues to store and retrieve messages.

When the client is sending the messages faster than the receiver receiving it or the client is sending the message when the receiver is not available. So it uses a queuing mechanism between the client and server to avoid the message being misplaced.

It is asynchronous mechanism where messages can be sent even though the receiver is not available

For eg Email system

Distributed Object Technology

The distributed object technology has changed the scope of middleware technologies to one step up where objects are distributed to the remote server to facilitate the client.

Eg RMI and CORBA

The distributed object mechanism hides the communication interfaces and their details to provide access to the remote object efficiently.

Remote Method Invocation

These objects are distributed and located by using the RMI registry.

The client can access remote objects by using the interfaces.

Disadvantage

It didn't support the concept of heterogeneity and is compatible with java platform only. Common object request broker architecture(CORBA)-

It is one of the most popular distributed object technologies where objects can be accessible from remote locations through ORB.

Server and client communicate with each other through object request broker bus.

To map the semantics of objects and fetch the appropriate object an interface definition language is used.

It is evolved with service based middleware where service models are used.

In the service model, the services are published by the service providers and consumed by the service consumer.

eg Service oriented Architecture (SOA)

➢ Services offered by middleware

Messaging middleware facilitates communications between distributed applications and services.

Object or ORB middleware enables software components or objects to communicate and interact with a program -- such as containers -- across distributed systems.

Remote Procedure Call (RPC) middleware provides a protocol that enables a program to request a service from another program located on a separate computer or network.

Data or database middleware enables direct access to, and interaction with, databases; it typically includes SQL database software.

Transaction or transactional middleware ensures transactions move from one phase to the next via transaction process monitoring.

Content-centric middleware allows client-side requests for specific content and abstracts and delivers it

Embedded middleware facilitates communication and integration between embedded apps and real-time operating systems.

API middleware enables developers to create and manage their application's APIs.

Asynchronous data streaming middleware enables data sharing between applications by replicating data streams in an intermediate store.

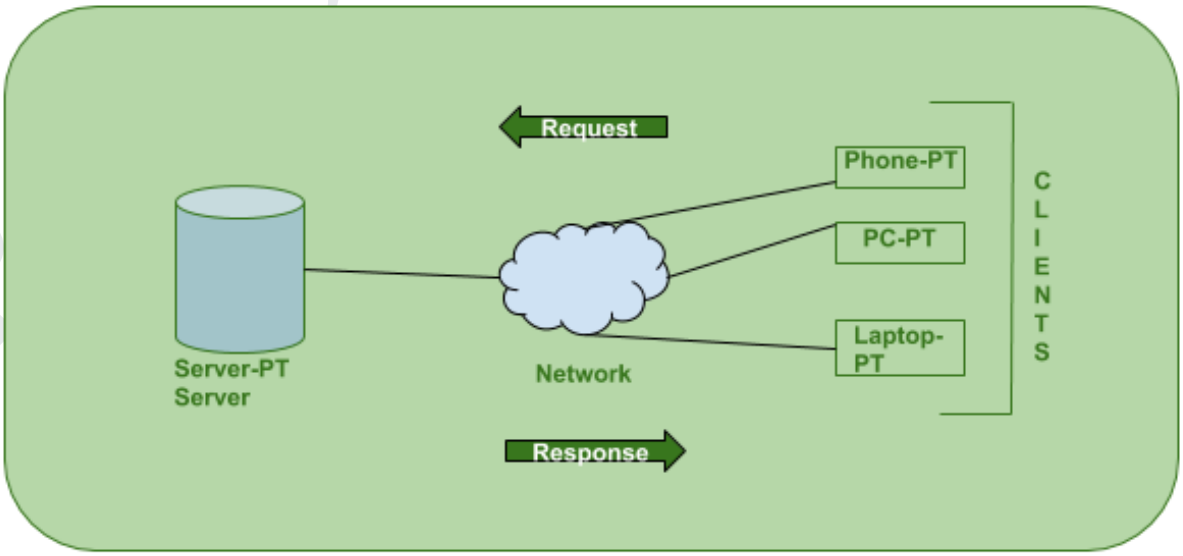
➢ Client Server model

The Client-server model is a distributed application structure that partitions task or workload between the providers of a resource or service, called servers, and service requesters called clients.

In the client-server architecture, when the client computer sends a request for data to the server through the internet, the server accepts the requested process and deliver the data packets requested back to the client.

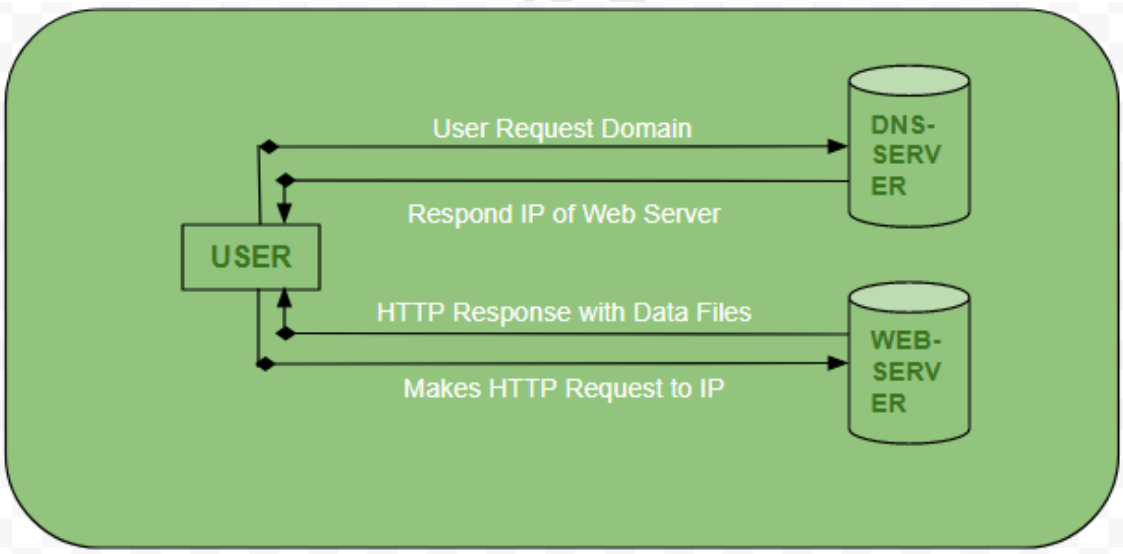
Clients do not share any of their resources.

Examples of Client-Server Model are Email, World Wide Web, etc.



How does the browser interact with the servers ?

- There are few steps to follow to interact with the servers and clients.
- User enters the URL(Uniform Resource Locator) of the website or file. The Browser then requests the DNS(DOMAIN NAME SYSTEM) Server.
- DNS Server lookup for the address of the WEB Server.
- DNS Server responds with the IP address of the WEB Server.
- Browser sends over an HTTP/HTTPS request to WEB Server's IP (provided by DNS server).
- Server sends over the necessary files of the website.
- Browser then renders the files and the website is displayed.



Types

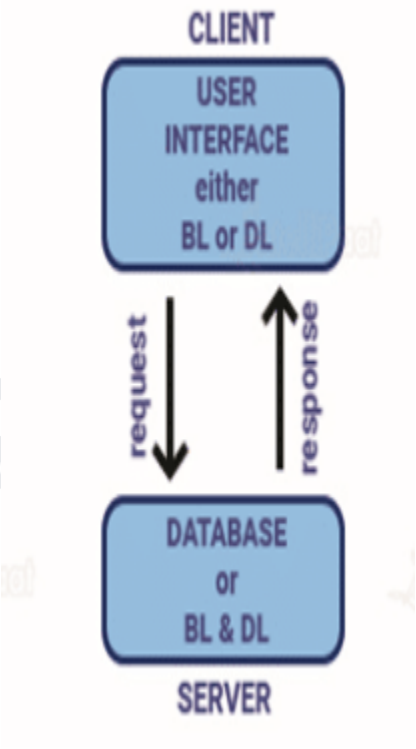
- 1 tier architecture
- 2-tier architecture
- 3-tier architecture

- N-tire architecture

1-tier architecture

Used to describe a specific type of software architecture in which all the necessary components for the functioning of an application are accessible under the same package.

2-tier architecture



The best environment is possessed by this architecture, where the client's side stores the user interface and the server houses the database, while either the client's side or the server's side manages the database logic and business logic.

3-tier architecture

Unlike 2-tier architecture that has no intermediary, in 3-tier client-server architecture, middleware lies between the client and the server.

If the client places a request to fetch specific information from the server, the request will first be received by the middleware.

It will then be dispatched to the server for further action. The same pattern will be followed when the server sends a response to the client.

The framework of 3-tier architecture is categorized into three main layers, presentation layer, application layer, and database tier.

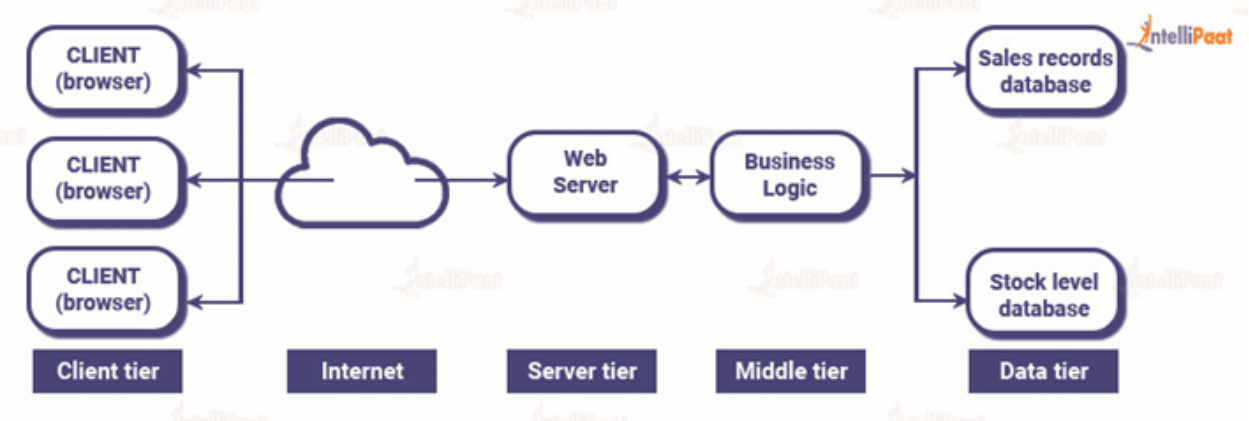
All three layers are controlled at different ends.

While the presentation layer is controlled at the client's device,

the middleware and the server handle the application layer and the database tier respectively.

Due to the presence of a third layer that provides data control, 3-tier architecture is more secure, has invisible database structure, and provides data integrity.

N-Tier architecture



N-tier architecture is also called multi-tier architecture.

It is the scaled form of the other three types of architecture.

This architecture has a provision for locating each function as an isolated layer that includes presentation, application processing, and management of data functionalities.