CLIENT SERVER ARCHITECTURE:

Client-Server architecture is an architectural deployment style that describe the separation of functionality into layers with each segment being a tier that can be located on a physically separate computer. They evolved through the component-oriented approach, generally using platform specific methods for communication instead of a message-based approach.

This architecture has different usages with different applications. It can be used in web applications and distributed applications. The strength in particular is when using this architecture over distributed systems. In this course work, I will furthermore invest this through the example of three-tier architecture in web applications.

Structure

Using this architecture the software is divided into 3 different tiers: ***Presentation tier, Logic tier, and Data tier***. Each tier is developed and maintained as an independent tier.

*1-**Presentation tier***

This is the topmost level of the application. The presentation layer provides the application's user interface (UI). Typically, this involves the use of Graphical User Interface for smart client interaction, and Web based technologies for browser-based interaction. The presentation tier displays information related to such services as browsing merchandise, purchasing, and shopping cart contents. It communicates with other tiers by outputting results to the browser/client tier and all other tiers in the network.

***2-Logic tier* (called also business logic, data access tier, or middle tier)**

The logic tier is pulled out from the presentation tier and, as its own layer; it controls an application's functionality by performing detailed processing. Logic tier is where mission-critical business problems are solved. The components that make up this layer can exist on a server machine, to assist in resource sharing. These components can be used to enforce business rules, such as business algorithms and legal or governmental regulations, and data rules, which are designed to keep the data structures consistent within either specific or multiple databases. Because these middle-tier components are not tied to a specific client, they can be used by all applications and can be moved to different locations, as response

time and other rules require. For example, simple edits can be placed on the client side to minimize network round-trips, or data rules can be placed in stored procedures.

3-Data tier

This tier consists of database servers, is the actual DBMS access layer. It can be accessed through the business services layer and on occasion by the user services layer. Here information is stored and retrieved. This tier keeps data neutral and independent from application servers or business logic. Giving data its own tier also improves scalability and performance. This layer consists of data access components (rather than raw DBMS connections) to aid in resource sharing and to allow clients to be configured without installing the DBMS libraries and ODBC drivers on each client. An example would be a computer hosting a database management system (DBMS), such as a Microsoft SQL Server database.

Components Interconnections:

3 tier application architecture is characterized by the functional decomposition of applications, service components, and their distributed deployment, providing improved scalability, availability, manageability, and resource utilization. During an application's life cycle, the three-tier approach provides benefits such as reusability, flexibility, manageability, maintainability, and scalability. Each tier is completely independent from all other tiers, except for those immediately above and below it. You can share and reuse the components and services you create, and you can distribute them across a network of computers as needed. You can divide large and complex projects into simpler projects and assign them to different programmers or programming teams. You can also deploy components and services on a server to help keep up with changes, and you can redeploy them as growth of the application's user base, data, and transaction volume increases.

Logic layer is moved outside the presentation layer and into the business layer as it enhances reuse. As applications grow, applications often grow into other realms. Applications may start out as a web application, but some of the functionality may later be moved to a smart client application. Portions of an application may be split between a web site and a web or windows service that runs on a server. In addition, keeping logic helps aid in developing a good design (sometimes code can get sloppier in the UI).

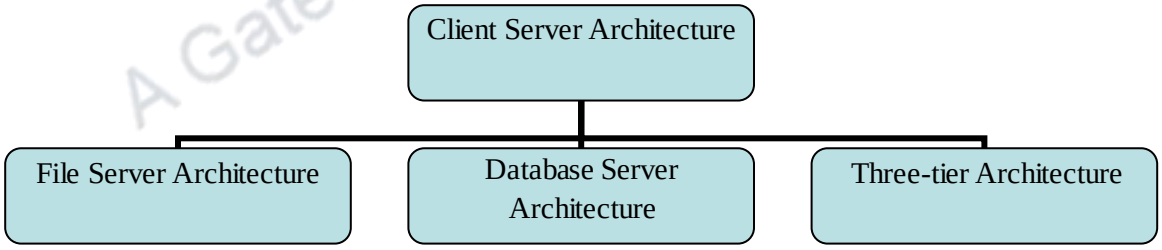
The main benefits of the 3-tier architectural style are:

- Maintainability. Because each tier is independent of the other tiers, updates or changes can be carried out without affecting the application as a whole.
- Scalability. Because tiers are based on the deployment of layers, scaling out an application is reasonably straightforward.
- Flexibility. Because each tier can be managed or scaled independently, flexibility is increased.
- Availability. Applications can exploit the modular architecture of enabling systems using easily scalable components, which increases availability.

Consider the 3-tier architectural style if the processing requirements of the layers in the application differ such that processing in one layer could absorb sufficient resources to slow the processing in other layers, or if the security requirements of the layers in the application differ. For example, the presentation layer should not store sensitive data, while this may be stored in the business and data layers. The 3-tier architectural style is also appropriate if you want to be able to share business logic between applications, and you have sufficient hardware to allocate the required number of servers to each tier.

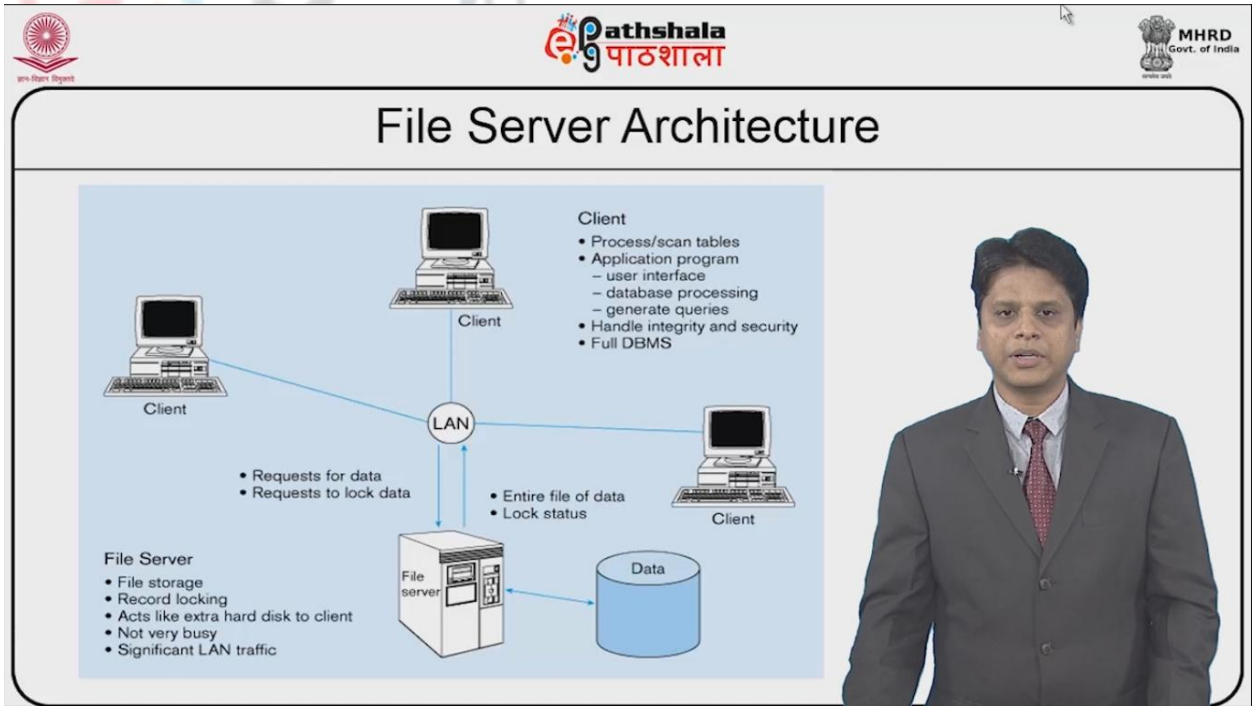
The client server architectures are of three types

1. Filer Server Architecture
2. Database Server Architecture
3. Three-tier Architecture



FILE SERVER ARCHITECTURE

The first client server architecture being developed is the file server architecture., all processing will be done at the PC that requested the data that is client handles the presentation logic, the processing logic and much of the storage logic. That means out of the hundred percent work that need to be done every logics that we have seen application logics that we have seen like presentation ;logic, processing logic and storage logic more than 80% of the job it will be carried out at the client side itself. So that is what I am trying to say here. All processing is done at the PC that requested the data that is client handles the presentation ;logic processing logic and much of the storage logic. A file server is a device that manage sonly the file operation and shared by each of the client PCs attached to the local area network. Each filer server acts as a n additional hard disk so wherever I am talking about a file server every file server will be ac5ring as an extra hard disk which will be present not only at the server level and which will also be present at the client side. For each of the client PC. Each PC maybe called a fat client because this client does an extensive work based on that I am calling this client as a fat client so most of the processing since it happens at the client side it is called fat client. Entire file are transferred from the server to the client for processing. So anything that the client has requested that query will be taken by the server and the server will respond all the possible states back to the server without any processing and client after the client receives all the service or the files requested for processing the client executes everything and it will be called a process.



So this is the complete architecture of a file server system where if you look at the machine all the clients are connected by a local area network to a file server this file server in turn manages all the data. So if you look at the major responsibility of a client and the responsibilities of the server, client does processing and scanning of tables, client does application program user interface not only that client does data processing, client generates queries, handles integrity that means it checks of the validation of the data and it also checks for the identification of from which node it has come from so it also have to handle security aspects of it. So that means almost all the processes taking at the client level so it maintains only a copy of DBMS that is available in the server. At the same side, if you look at the file server architecture the file server looks after the storage of all the files and locking of records acts like an extra hard disk to the client and this will not be very busy because many a times I am not transferring all the information from the server's end I am transferring the information to the client and I don't process the information. So it is not very busy and it has got significant LAN traffic. So when you look at the entire architecture although many machines are connected to the server the responsibility of the serve r will any request that I get form different client machine is push all the data from my end to the client's side. It is a responsibility of a client to take up all the data that is available in the server process whatever which is required for the client to execute those will be done at the client by itself. So if you look at the networking system the client can request for a data, request to log the data and from the server it will be given back where the entire file of the data will be transferred to the client and the lock status will also be given to the client. Say lock like read, write locks once the client request it means it is trying to read from the server so it locks. so whatever information that is to be read, maybe it could be put as a shared lock or an exclusive lock where in case of a shared lock many clients can request to a server, what are the information that the client request to read from the server and if it is to be an exclusive lock if it is from the client's end it will be locked at the server then the client after transferring all the information to the client then the lock will be released. So those information will be handled at the server's end. So as I have already told you this filer server architecture since most of the work is carried out at the client end, that is out of 100% of the job 80% of the work is being done at the client's end, so I call these type of machine as fat client machines.

Every system will have its own drawbacks. There are a few drawbacks wherein case because

of transfer of information from the server to the client as requested there will be a huge amount of data transfer on the network because when an client wants to access a whole data table, in case a client wants a whole data table to processing a simple information then the whole data will be transferred from the server to the client. So this has to be transferred to the PC so the server is doing very little work that means serve locate where the table is transfers the entire table to the entire table to the client machine. So server does only little work, network is transferring large blocks of data and the client is being busy processing all the data with an extensive data manipulations. The second issue that raises because of the file server architecture, each client is authorized to use database management system, when a DB program runs particularly on that PC. Thus there is one database but many concurrently running copies of DBMS. The DBMS copy I each client PC must manage the shared database integrity. So programmers must be sophisticated to recognize various subtle conditions that can arise in a multiple user database environment.

DATABASE SERVER ARCHITECTURE:

It is similar to data warehouse where the website store or maintain their data and information. A Database Server is a computer in a LAN that is dedicated to database storage and retrieval. The database server holds the Database Management System (DBMS) and the databases. Upon requests from the client machines, it searches the database for selected records and passes them back over the network**.**

A database server can be defined as a server dedicated to providing database services. Such a server runs the database software. A database server can typically be seen in a client-server environment where it provides information sought by the client systems.

A database server is useful for organizations that have a lot of data to deal with on a regular basis. If you have client-server architecture where the clients need process data too

frequently, it is better to work with a database server. Some organizations use the file server to store and process data. A database server is much more efficient than a file server.

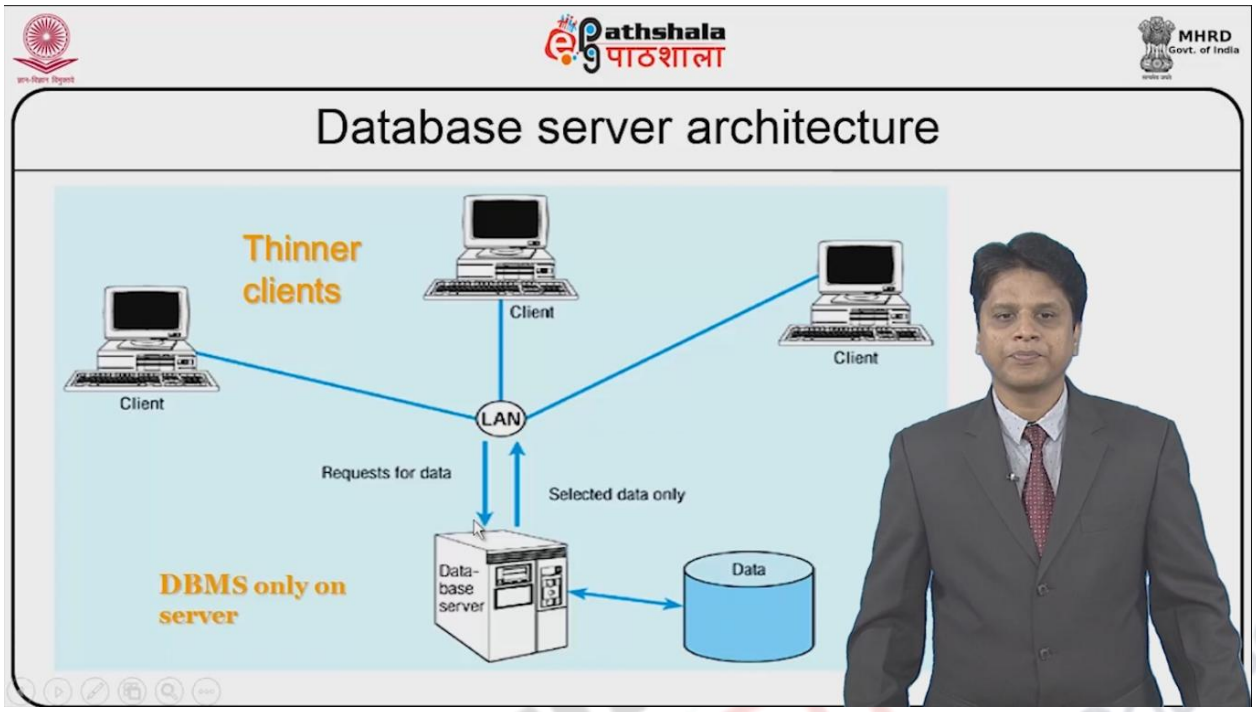
In Database Network the client execute SQL requests to the database server. The Network Database Server Process the client database request and the executed answers of SQL command are come back over the network computer. In the whole concept Database server serves its own power to process the request or search the requested result. The Database server some time also known as SQL engine.

All database functions are controlled by the database server. Any type of computer can be used as database server. It may be microcomputer, minicomputer or mainframe computer. In large organization networks, the mainframe computers are used as server.

Some people refer to the central DBMS functions as the back-end functions, whereas the application programs on the client computer as front-end programs. You can say that client is the application, which is used to interface with the DBMS, while database server is a DBMS.

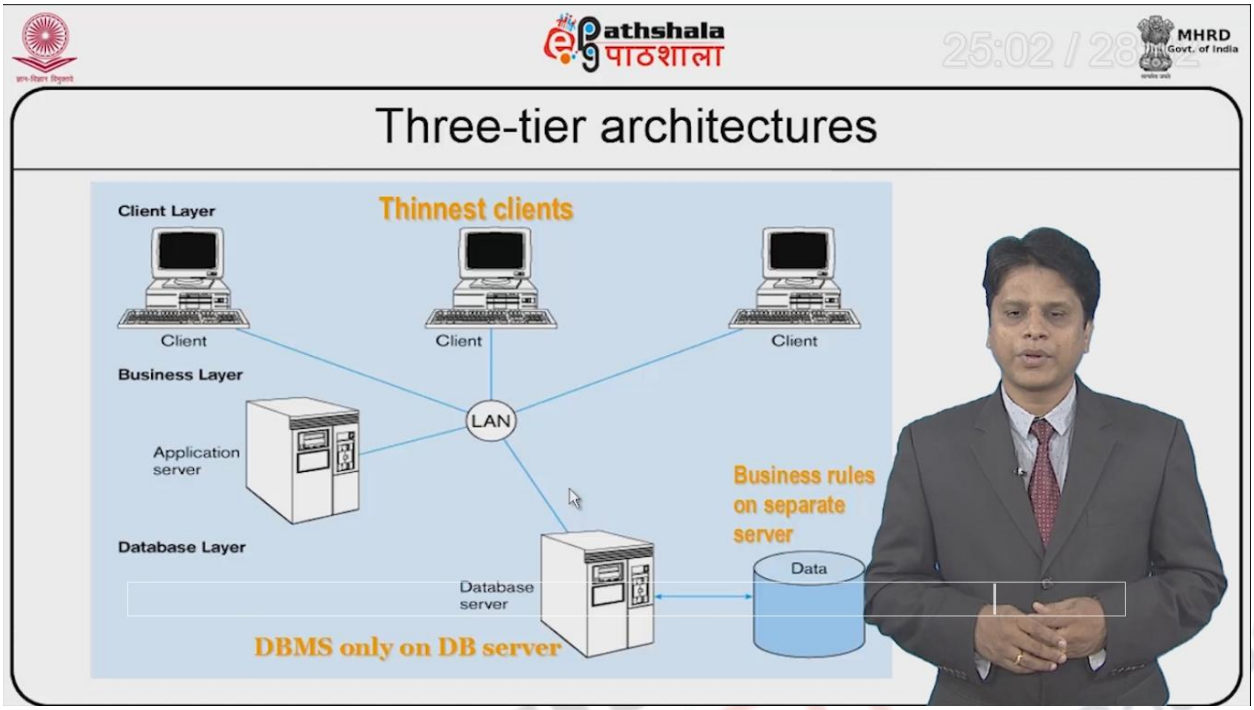
The Database server manages the recovery security services of the DBMS. It enforces the constraints that are specified inside the DBMS. It controls and manages all the clients that are connected to it. It handles all database access and control functions.

It provides concurrent access control. It provides better security and server hides the DBMS from clients. It provides the multi-user environment. Several users can access the database simultaneously. All the data is stored on the data server therefore, the DBA can easily create the backup of the database.



A standard called ***ODBC (Open Database Connectivity)*** provides an application programming interface (API), which allows client side programs to call the DBMS on the server side. For this purpose, necessary software should be installed on both sides (i.e., client and server). Hence, a client program connects to the Database server and sends requests (queries) using the ODBC Application Programming Interface (API). The-server processes the queries and sends back the results of queries to the client program, which are processed by the client computer

THREE TIER ARCHITECTURE



1. Presentation Layer

In the presentation layer the user interaction takes place as defined before. The user enters the address in the web browser and in the browser the URL is decoded into protocol/host/file, i.e. host name converted to IP address. Then an issue request is sent to remote server using appropriate protocol (usually HTTP).

Also a returned HTML from the logic tier might be accepted. In the presentation layer interaction with client side scripts (e.g. using DHTML) is supported and user inputer of variety controls on the form are accepted.

2. Logic Tier

In the logic Tier the application's functionality is done by performing detailed processing of data from presentation layer. Server such as Appache (or IIS) or Server Script (such as PHP) can be used to support this.

With Server (Apache or IIS) the appropriate action to be taken is identified, such as fetching a file, or passing request to an interpreter. Also it sends an output back to caller in MIME package. As such support for thousands for concurrent users, multithreading (allow

multiple processor to run concurrently) and caching (holding results in a temporary store to reduce recalculation) is achieved).

With Server script (example in PHP) interacting with server such as accessing input or generating input is done. It interprets the requests according to business rules and past transactions from this client, and requests appropriate data from the persistence layer. It also computes the derived data and creates HTML (or GIF…) for the page.

3. Data Layer

This tier consists of database servers. The interaction with the database is done using standard languages such as SQL queries using database specific protocol over TCP/IP. The data structures (for example tables) are defined and modified themselves, that insertion, updating and deleting of data for example. Data maintenance should be maintained with backup and recovered. Access to compilation of queries should be optimized, with indexing or replication of tables. An example of technology using this would be .NET that is built into the .NET framework, as ADO.NET contains a mechanism to query data out of the database and return it to the caller in a connected or disconnected fashion.

Real life example of a web system explained above would be in Emails done using 3 Tier Architecture. Reading e-mail using a Web-based interface, such as Hotmail, uses a three-tier architecture. The three tiers are:

1. Presentation Layer: The client's web browser that sends HTTP requests to the Web server.
2. Logic Layer: The Web server:
    a. sends HTTP responses to the Web client
    b. Translates the client's HTTP requests into SMTP packets which are then sent to the
    Mail server.
3. Data Layer: The Mail server performs the following functionality, and when performed it is
    transformed to the Logic Layer.
    a. When completed, an e-mail message is sent by the sender's e-mail client as an

SMTP packet to the local mail server.

b. The mail server's message transfer agent next reads the packet's destination address
and sends it over the Internet to the receiver's mail server.

c. The destination mail transfer agent then stores the message in the receiver's mail
box.

d. When the receiver next accesses e-mail, his or her user agent contacts the local mail
server which then downloads the message to the receiver's client computer.