A · P · U

ASIA PACIFIC UNIVERSITY
OF TECHNOLOGY & INNOVATION

CLIENT-SERVER ARCHITECTURE
HOW IT WORKS.
INDIVIDUAL ASSIGNMENT

TITLE: CLIENT-SERVER ARCHITECTURE

LECTURER NAME:	Mrs. Sathiapriya A/P Ramiah
STUDENT NAME:	Mrisho Abeid Omary
STUDENT ID:	TP033289
INTAKE CODE:	UCDF1310 (BIT)
ISSUED DATE:	02-Sep-2014
SUBMITION DATE:	17-Oct-2014
WORD COUNT:	3000 Words

TABLE OF CONTENTS:

TABLE OF FIGURES:

0.0 ABSTRACT:

This study aims at defining the Client-Server Architecture. Through research from other articles, the internet, books and lecture notes, this report will cover on the general meaning of client and server computers, with their characteristics, characteristics of a client-server architecture, middleware, the types of the client-server architecture and the advantages and disadvantages of the client-server architecture. It is also includes a conclusion, appendix/appendices and references for the topic in mind.

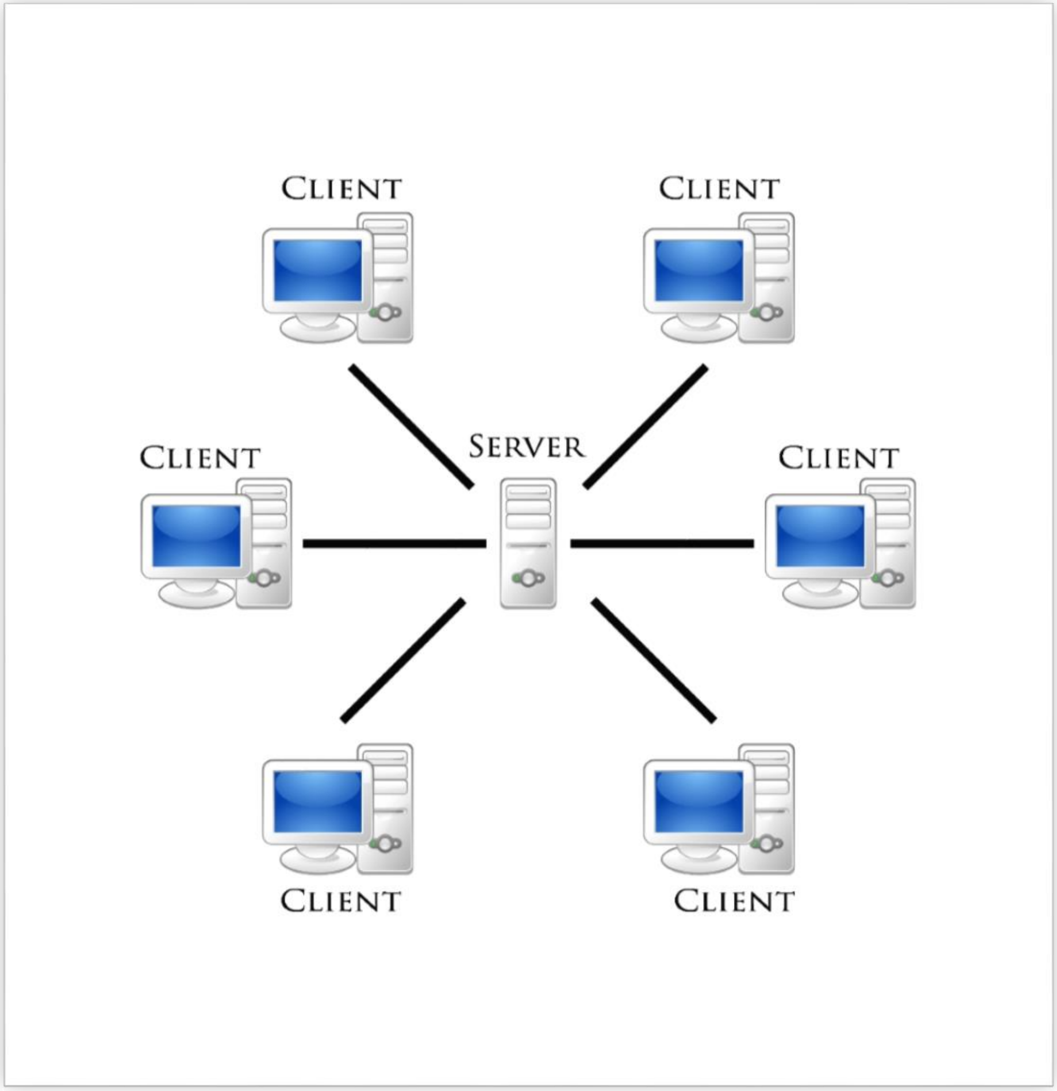


FIGURE 1: CLIENT-SERVER ARCHICTECTURE

(*Source: Google Images.com, 2014*)

CLIENT-SERVER ARCHITECTURE

HOW IT WORKS.

Mrisho Abeid Omary| TP033289 | INTERNET APPICATIONS | October 17, 2014

1.0 INTRODUCTION:

The information processing is divided into unique parts. A part is either request (client) or provider (server). The client sends amid the information processing one or more demands to the servers to perform pointed out errands. The server part gives administrations to the clients. (*See figure below*) (*Source: Client/Server Architecture*)

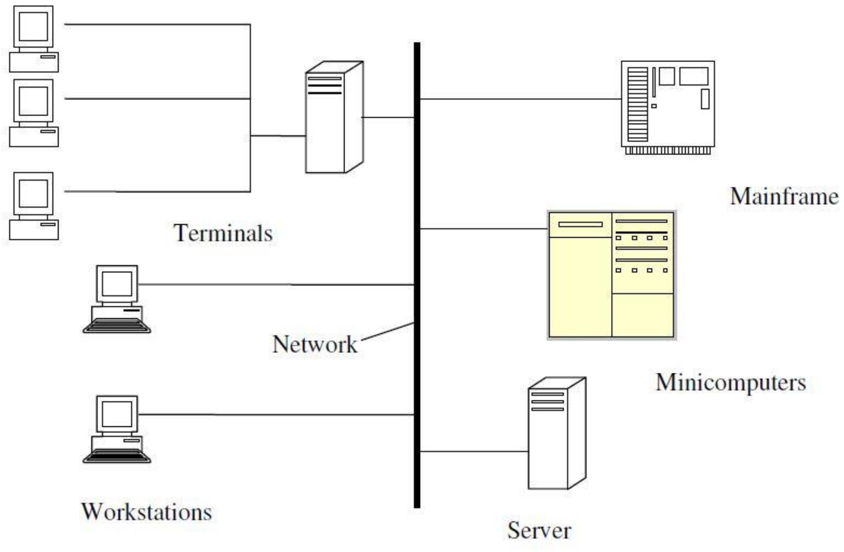


FIGURE 2: CLIENT-SERVER MODEL

1.1 CLIENT COMPUTER:

Client - is a solitary workstation that gives presentation services and the proper computing, integration and the database services and the interfaces important to the business need.

1.1.1 TYPES OF CLIENTS:

There are several types of clients. The following are some of the client computers in a client-server architecture;

- Thin Client
- Fat Client
- Dumb Client

1.1.2 CHARACTERISTICS OF CLIENTS:

- It is invoked locally by users, and executes just for one time.
- Runs locally on the user's personal computer.
- Actively initiates contact with the server.
- It can access many services as required, but actively contacts one remote server at a go.
- It doesn't need special hardware or complex operating systems.
- Waits to receive replies from the server.

1.2 SERVER COMPUTER:

Server- is one or more multi-client processors with shared memory providing integration and database administrations and the interfaces applicable for the business need.

1.2.1 TYPES OF SERVERS:

There are several types of server. The following are some of the common servers in a client-server architecture;

- DNS Server (*Domain Name Server*)
- Mail Server
- Application Server
- Web Server
- Database Server
- File Server
- FTP Server (*File Transfer Protocol*)
- Object Servers

1.2.2 CHARACTERISTICS OF SERVERS:

- Runs on shared computers.
- Passively waits for contact from remote clients.
- They can connect many computers at the time.
- Accepts contact from clients but offers one service at a time.
- Needs powerful hardware and sophisticated operating systems.

1.1 CHARACTERISTICS OF A CLIENT-SERVER ARCHITECTURE:

1 **Service-** the relationship between a client and server is primarily that which runs on different machines. The server process is the provider while the client is the consumer of the service.

2 **Shared resources**- Servers can service many client computers at the same times and regulate their access to shared resources.

3 **Mix –and-match**- client/server ideal software are independent of operating system software or hardware platforms.

4 **Scalability**- they can be either horizontally or vertically. Horizontal scaling is the adding or removing of client computers with just a slight performance impact, while vertical scaling is changing to a larger and much faster machine or multi-servers.

5 **Integrity**- the serve’s code and data are stored centrally, which results to cheaper maintenance and the guarding of shared data integrity, while at the time clients remain personal and independent.

6 **Asymmetrical protocols-** clients and servers have a *many-to-one* type of relationship. *Initiating* of a dialog starts from the client by requesting a service, while are passively waiting for the client requests.

7 **Transparency of location-** the server process can run on the same machine as the client or a different one across the network. The client/server software often hides the location of the server from the clients by forwarding their service calls when needed. In essence, a program can be a client, server, or both.

1.2 TYPES OF CLIENT-SERVER ARCHITECTURES:

There are three types of client-server architectures which are categorized by "*tiers*". The following are descriptions of each, with their characteristics, components, layout and advantages/disadvantages;

1.2.1 THE 2-TIER CLIENT-SERVER ARCHITECTURE:

In this particular tier, the applications can be divided into logical functional components such as;

- Presentation Services
- Presentation Logic
- Business Logic
- Distribution Logic
- Distribution Services
- Database Logic
- Database Services
- File Services

These functions are distributed in the network where, some run on the client computer the others on the server. (*See diagram below*) (*Source: Client/Server Architecture*)

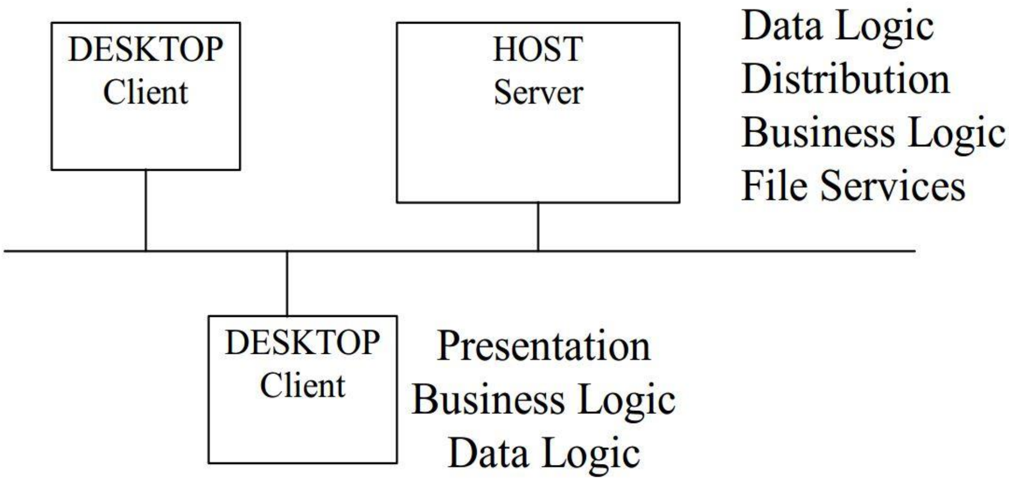


FIGURE 3: FUNCTIONAL DISTRIBUTIONS IN A 2-TIER ARCHITECTURE

As shown in the picture above, Presentation, and Data Logic run on the *client's* computer and Data Logic, Distribution and File Service run on the *server's* computer while Business Logic runs on both *client* and *server*.

1.2.1.1 CHARACTERISTICS OF A 2-TIER ARCHITECTURE:

The following are some of the characteristics of the 2-tier client-server architecture;

- The 2-tier architecture helps to improve *usability* by supporting a form-based and user friendly interface and scalability by adding up to 100 user.
- It is also flexible by allowing data to be shared easily.
- It requires minimal operator intervention and is frequently used in non-complex, non-time critical information processing systems.
- It consists of three components distributed in two layers which are *client (*requester of services*)* and *server (*provider of services*)*.

  The three components are;

  1. User System Interface.
  2. Processing Management.
  3. Database Management.

- Applications can be built quickly by using *visual builder tools;* which can be used for developing applications for decision support systems or small-scale groupware.
- The application logic in a 2-tier architecture is hide inside the user interface on the client's computer or within the database in the server on both sides.

The following picture shows the 2-tier Client-Server Architecture; (*See figure below*) (*Source: Client/Server Architecture)*

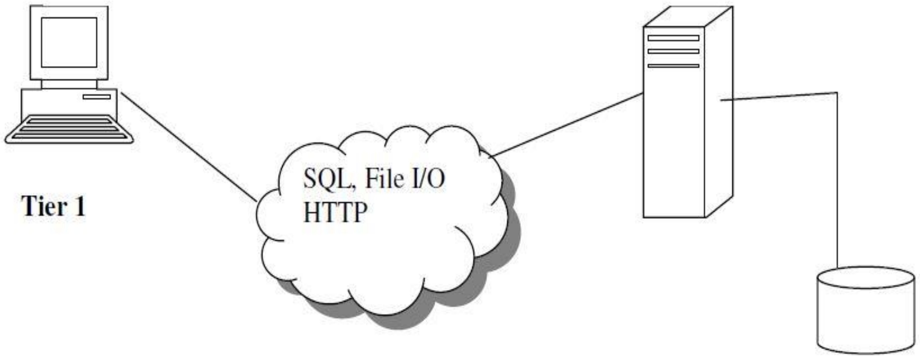


FIGURE 4: 2-TIER CLIENT-SERVER MODEL

1.2.2 THE 3-TIER CLIENT-SERVER ARCHITECTURE:

In this particular tier, the distribution of different functional components can be differentiated in 3 different sites, such as;

- 1st tier: Presentation Logic- it runs on the client's computer.
- 2nd tier: Business Logic- it is the application server.
- 3rd tier: Database Logic- it runs in the database server.

(*See figure below*) (*Source: Google Images.com*)

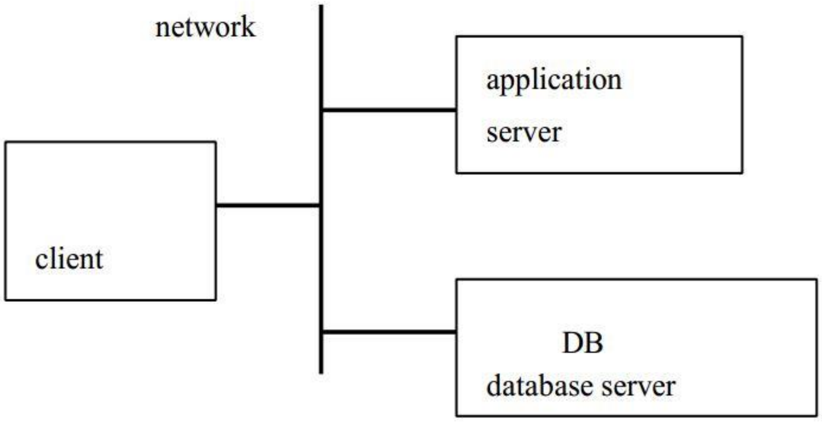


FIGURE 5: FUNCTIONAL COMPONENTS IN A 3-TIER ARCHITECTURE

1.2.2.1 CHARACTERISTICS OF A 3-TIER ARCHITECTURE:

The following are some of the characteristics of the 3-tier client-server architecture;

- Less software is stored on the client making it much faster.
- The application logic is (*or process*) is lives in the middle tier separated from the data and user interface.
- There is increased security since the data/information is distributed in the servers.
- The cost for supporting a 3-tier architecture is low due to distributed functionalities in the network.
- Scalability is much higher and there is more efficiency and power in this particular architecture.
- The structure of a 3-tier architecture is more complex making it harder to maintain.
- There is a big problem in heterogeneous data sources since duplication can easily happen due to storing information in different servers.
- The 3-tier architecture cam be an *internet based architecture* where there is an open and standardized interface. (*See figure below*) (*Source: WordPress.com, 2014*)

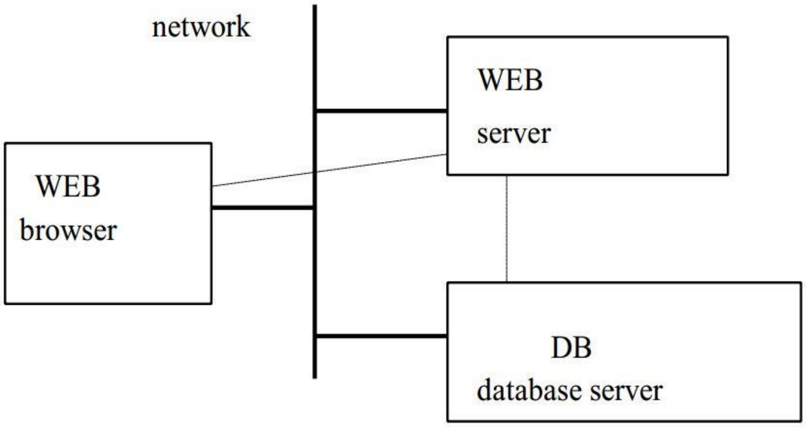


FIGURE 6: INTERNET BASED 3-TIER ARCHITECTURE

In this particular 3-tier architecture (*See figure 6 above*), the *Web Servers* can directly access the databases through CGI (*Common Gateway Interface*) interface.

The following is a diagram elaborating or showing the layout of the components in a 3- tier architecture; (*See figure below*) (*Source: Progress.com, 2014*)

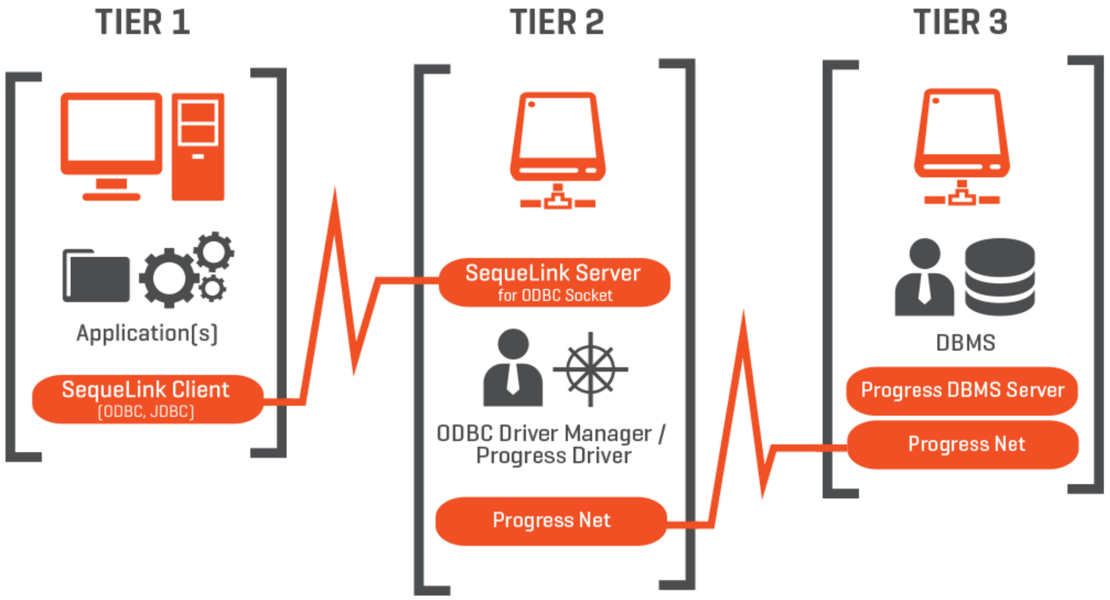


FIGURE 7: THE 3-TIER CLIENT-SERVER ARCHITECTURE

1.2.2.2 WHEN TO USE A 3-TIER ARCHITECTURE:

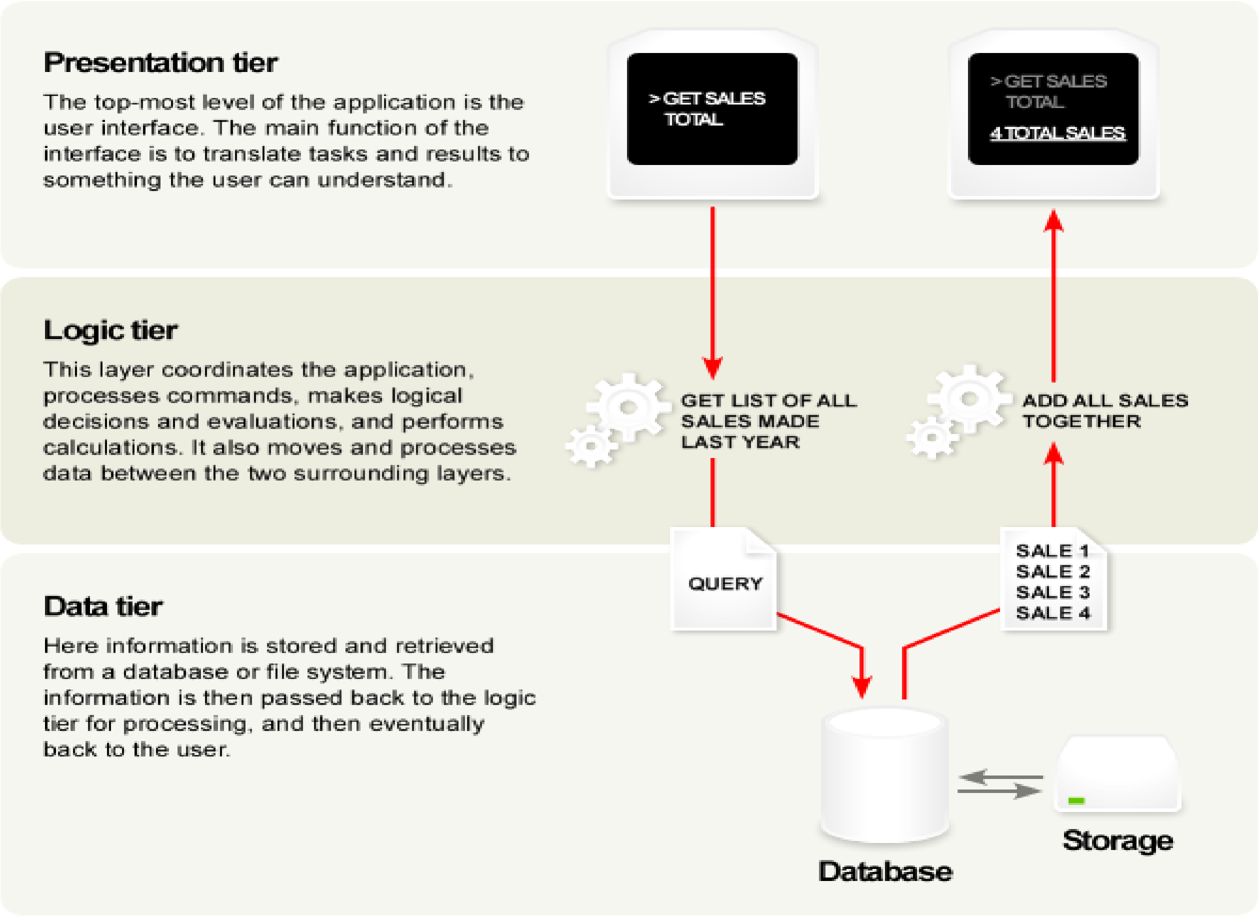
As per Gartner Group's for more smaller tasks , 2-tier applications are less demanding to create, than 3-tier, however as the application gets more complex and hard, 2-tier applications get to be exponentially harder to develop.

Due to these reasons, Gartner's Group suggests a 3-tier architecture to be implemented instead of a 2-tier if the application has any of the following qualities;

- If there are more than 50 applications services/classes.
- The application is designed in different languages or created by different organizations.
- Applications implemented are to last more than 3 years especially when you expect modifications or additions.
- If there is a high workload of more than 60,000 transactions are to be made in a day or more than 300 concurrent users on the same system using the same database.

- If there are two or more different data sources like two DBMS (*Database Management System*) or a DBMS and a file system.
- Substantial communication between applications including inter-enterprise communication like *Electronic Data Interchange* (*EDI*).

A detailed explanation on the three layers involved in a 3-tier client-server architecture are elaborated in the picture below; (*See figure 8*) (*Source: Internet Applications Lecture 02- Page 13*)



Presentation tier

The top-most level of the application is the user interface. The main function of the interface is to translate tasks and results to something the user can understand.

Logic tier

This layer coordinates the application, processes commands, makes logical decisions and evaluations, and performs calculations. It also moves and processes data between the two surrounding layers.

Data tier

Here information is stored and retrieved from a database or file system. The information is then passed back to the logic tier for processing, and then eventually back to the user.

FIGURE 8: ELLABORATION OF THE 3 LAYERS IN A 3-TIER ARCHITECTURE

1.2.3 COMPARISON BETWEEN 2-TIER AND 3-TIER ARCHITECTURES:

	2-TIER	**3-TIER**
System Administration	Complex *(there is more logic on the client side)*	Less complex *(the application can be managed centrally on the server)*
Security	Low level of data security	High *(perfected at the service, method or object levels)*
Encapsulation of data	Low *(tables of data are visible)*	High *(clients summons services or methods)*
Performance	Poor *(a lot of SQL statements are sent over the network)*	Good *(only service request and responses are sent)*
Scale	Poor	Excellent
Application reuse	Poor *(colossal application on client)*	Excellent *(it can reuse services and objects)*
Ease of development	High	Getting Better *(clients can be created using standard tools, and some can be used to create both client and server sides of applications)*
Heterogeneous database support	None	Yes *(applications in 3-tier can use many databases inside the same business transaction)*

1.3 BUILDING BLOCKS IN A CLIENT/SERVER ARCHITECTURE:

Building blocks of a client/server architecture are the layouts in how it can be created. It is very simple and it works very well with the technologies that we have today. It is perfectly suitable for working with the needs of a *post-scarcity computing* world, where the client/server becomes the major standard for sharing and cooperating. The three main building blocks in a client server architecture are *client, middleware and server*; (*See figure below*) (*Source: Orfali, Harkely and Edwards, 1999*)

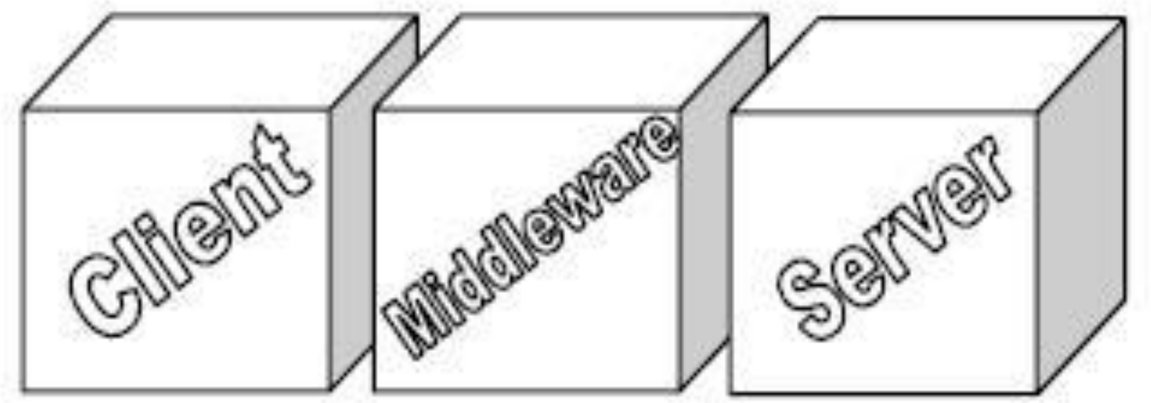


FIGURE 9: THE CLIENT/SERVER BUILDING BLOCKS

The next few paragraphs will elaborate and illustrate how the building blocks provisions are used in four different aspects;

1.3.1 TYPES OF BUILDING BLOCKS OF A CLIENT/SERVER ARCHITECTURE:

1. ***Client/Server Architecture for Tiny Shops and Nomadic Tribes:*** This implementation in this building block runs the client, the middleware software, and most business services on the same machine. (*See figure below*) (*Source: Orfali, Harkely and Edwards, 1999*)

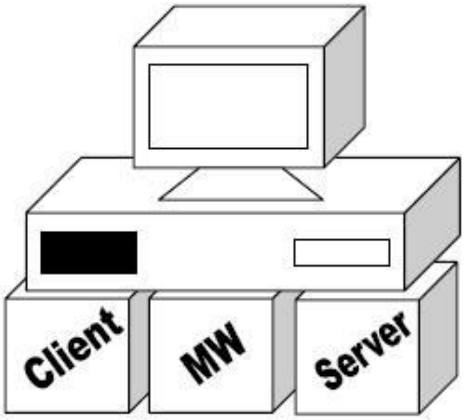


FIGURE 10: CLIENT/SERVER BUILDING BLOCK FOR TINY SHOPS AND NOMADIC USERS

2. ***Client/Server Architecture for Small Shops and Departments:*** This is a classic Ethernet client/single-server building block implementation. It's the predominant form of client/server today. (*See figure below*) (*Source: Orfali, Harkely and Edwards, 1999*)

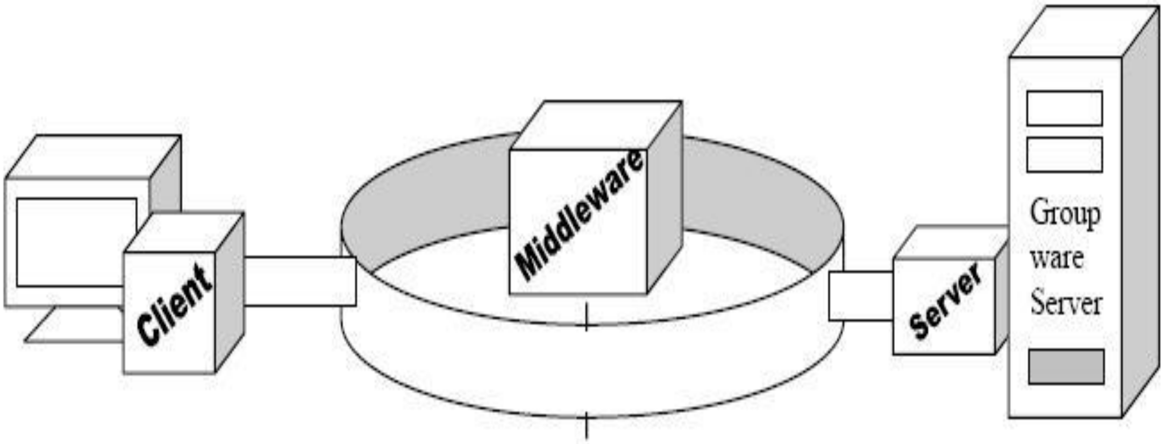


FIGURE 11: CLIENT/SERVER BUILDING BLOCK FOR SMALL SHOPS AND DEPARTMENTS

3. ***Client/Server Architecture for Intergalactic Enterprises:*** This implementation is a multi-server building block. The servers in this block present a client with a single-system image. They can be spread all over the organization, but also can be made to appear like they're part of a local desktop. (*See figure below*) (*Source: Orfali, Harkely and Edwards, 1999*)

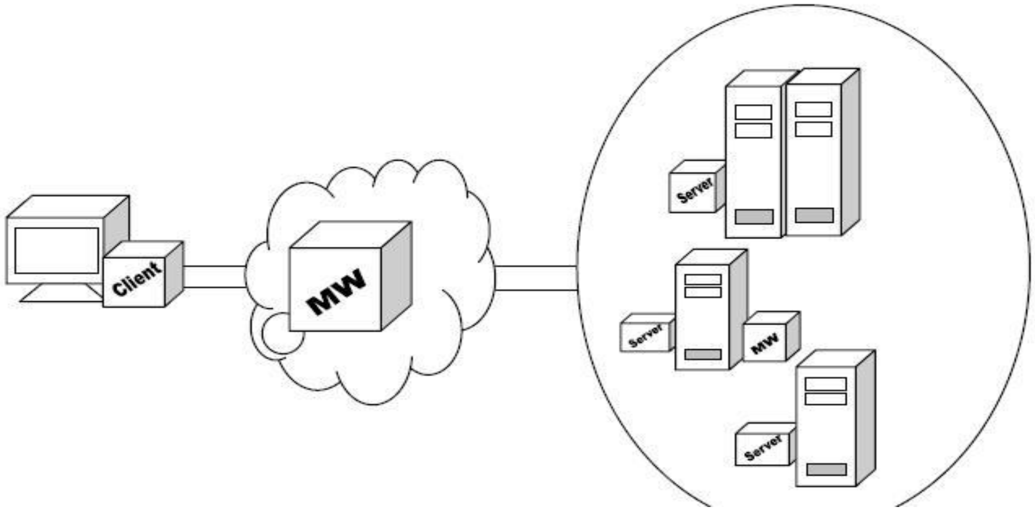


FIGURE 12: CLIENT/SERVER ARCHICTURE BUILDING BLOCK FOR INTERGALACTIC ENTERPRISE

4. ***Client/Server Architecture for a Post-Scarcity World:*** This transforms every machine in the network into a client and a server. All negotiations on every machine will be handled by their personal agents with their peer anywhere in the universe. (*See figure below*) (*Source: Orfali, Harkely and Edwards, 1999*)

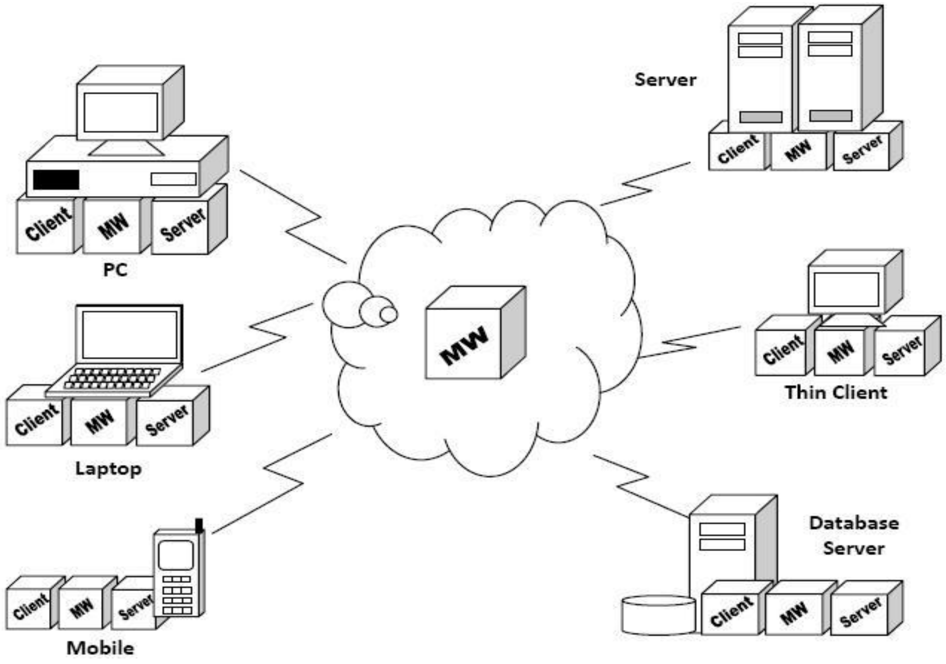


FIGURE 13: CLIENT/SERVER ARCHITECTURE BUILDING BLOCK FOR A POST-SCARCITY WORLD

1.3.2 CATEGORIES OF CLIENT/SERVER BUILDING BLOCKS:

The following are the categories of Client/Server Building Blocks which are "*Client/Server Software Infrastructure*" and "*Server/Server Infrastructure*";

1.3.2.1 CLIENT/SERVER SOFTWARE INFRASTRUCTURE:

1 ***The client building block***- this runs on the OS (*Operating System*) on the client's side of the application which provides a GUI (*Graphical User Interface*) or an OOUI (*Object Oriented UI*) that can access services distributed in the application. (*See figure 14 below*) (*Source: Orfali, Harkely and Edwards, 1999*)

2 ***The server building block***- this runs on a *shrink-wrapped server software* package on the server side of the application. The SQL servers, TP monitors, groupware servers, and the web are the five contending platforms for creating the next generation client/server architecture. The server depends on the operating OS for interfacing with the middleware in order to receive the requests for service from the client. (*See figure 14 below*) (*Source: Orfali, Harkely and Edwards, 1999*)

3 ***The middleware building block***- this runs on both the client and server sides of the application. This block is sub-categorized into three parts namely: *transport stacks*, NOSs (*Network Operating Systems*) and *service-specific middle ware*. (*See figure 14 below*) (*Source: Orfali, Harkely and Edwards, 1999*)

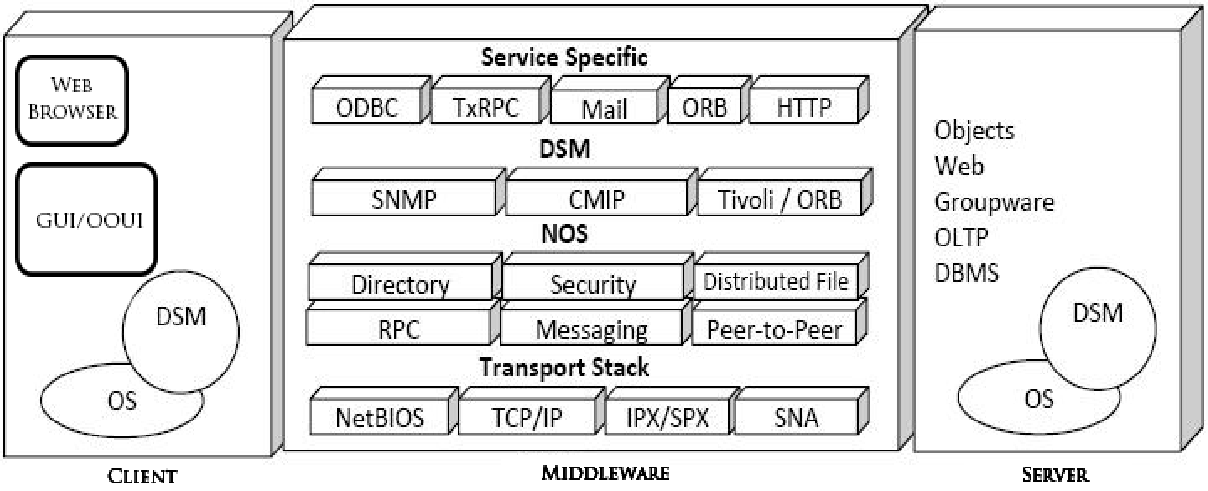


FIGURE 14: CLIENT/SERVER SOFTWARE INFRASTRUCTURE

1.3.2.2 SERVER/SERVER INFRASTRUCTURE:

In this particular building block, the middleware does not include the software that provides the actual services. However, it contains the software that is used to manage interactions between servers, this is also known as an *inter-server interaction*. The communication between a two servers is usually a *client/server interaction* in nature, this means servers are clients to other servers. However, these interactions between servers require specialized middleware to complete the process. For-instance, *a two phase commit protocol* can be used to synchronize a transaction that executes on multiple servers. (*See figure 15 below*) (*Source: Orfali, Harkely and Edwards, 1999*)

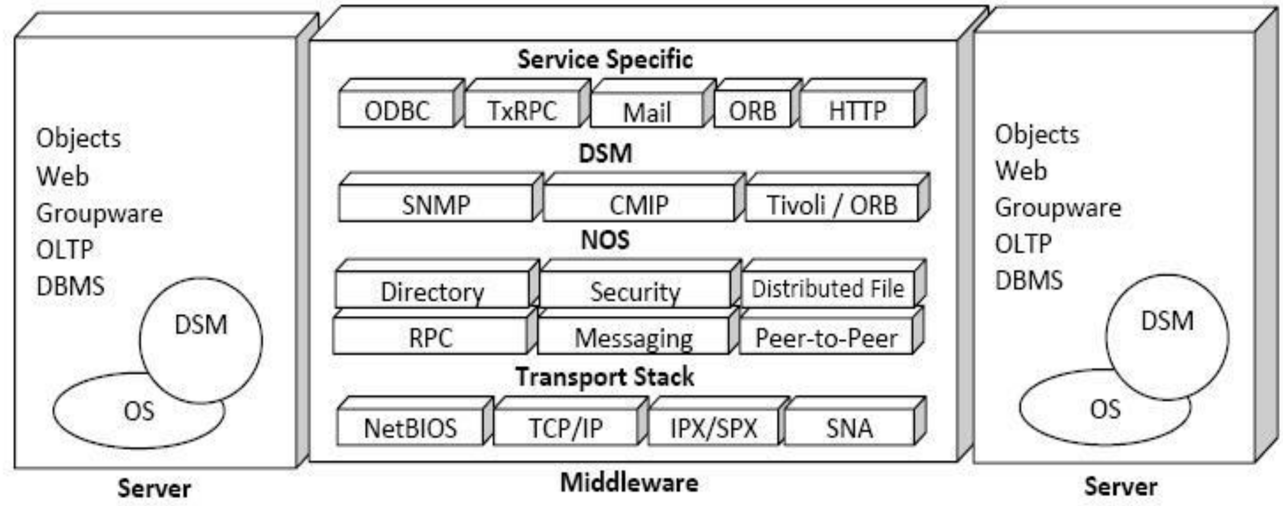


FIGURE 15: SERVER/SERVER INFRASTRUCTURE

1.4 ADVANTAGES & DISADVANTAGES OF A CLIENT/SERVER ARCHITECTURE:

There are several advantages and disadvantages in the client/server architecture.

1.4.1 ADVANTAGES OF CLIENT/SERVER ARCHITECTURE:

The following are some of the advantages of a client/server architecture;

1 **Enhanced Data Sharing-** The data in this structural design is collected as part of the normal business procedure and sustained on a server. SQL (*Structured Query Language*) is used to describe and handle the data which provides help for open access from all client computers.

2 **Integrated Services-** The client id only allowed to access the information in the desktop. There is no need to change terminal modes or to log into another processor in order to access the information.

3 **Data Interchangeability and Interoperability-** With use of SQL, vendors are enabled to develop production-class database engines to manage data in tables. Systems today that use SQL are fundamentally transparent to data storage sites and the technology of the data storage platforms.

4 **Independence of Location and Data Processing-** The syntax that was used for access was unique for each platform. Error messages, function keys, security, navigation methods, performance and editing were all visible. But today's users expect a good "*look and feel*".

5 **Encapsulation-** Is referred to as a capability of a network architecture to allow the roles and tasks of a computing system to be spread out among several independent computers known to each other in the network.

6 **Security-** Security is ensured to be high since all the data and information are stored in the server were only the network administrators can access or change it. This leaves a very small room for the client computers to take information from the network.

1.4.2 DISADVANTAGES OF A CLIENT/SERVER ARCHITECTURE:

The following are some of the disadvantages of the client/server architecture;

1 **Traffic Congestion**- As a number of simultaneous requests from the clients are supplied to the server, the workload can be too much causing an overload in the number of tasks being processed hence the network will lag or run slowly.

2 **Lack of robustness**- The paradigm in a client/server architecture lacks robustness in such a way that, if there should be a severe/critical failure in the server then the clients' requests will not be fulfilled.

3 **Time consuming**- If a large number of clients send requests to the server at the same time, the processing speed of the server is likely to run slow causing a delay in the processing of the functions or requests from the clients.

1.5 CRITICAL ANALYSIS:

According to the facts and explanations presented above, this research proves that in the matter of acceptance between the *2-tier* & *3-tier client/server architectures* the 3-tier is the best suited to handle most situations in the networking field. Due to the vivid comparison in the table above (*See page 10*), between the “two tiers”, the information clearly shows that the 3-tier architecture is the best option in terms of *security*, *performance*, *system administration*, *scaling*, *reusing of applications* & *supporting heterogeneous databases* and many more.

In this case, it is best advised to implement a *3-tier client-server architecture* rather than a 2-tier because it is *easier to use*, *implement*, *maintain*, *scale* and also *monitor*.

1.6 CONCLUSION:

To sum up, the client/server architecture is a well-known and most recognized network model which is applied nowadays. It offers more scalability, flexibility, usability, interoperability which improves the efficiency of the network. Even though there can be some problems with system and it's tools, solving them is very easy due to a more common and easy to use GUI. As discussed in the report, there are some issues that may occur due to certain reasons, but weighing the problems to the advantages, the advantages unquestionably rule-out the problems. With the facts of information mentioned above, it is safe to say that the *Client/Server Architecture* is indeed the best network architecture to use.

1.7 REFERENCES:

1 Orfali, R., Harkey, D., Edwards, J., ‘The Essential Client/Server’, ‘Survival Guide Third Edition’, pp. 11-35.

2 Orfali, R., Harkey, D., Edwards, J., ‘The Essential Client/Server’, ‘Survival Guide Second Edition’, pp. 17-43.

3 Client/Server Architecture (2013), ‘Client/Server Architecture: Course Material’, Client/Server Architecture, pp. 6-12.

4 Client/Server Systems (1990). ‘Introduction to Client/Server’, The Systems in a Client/Server Architecture, pp. 5-10.

5 Salem (1992), ‘The Complete Guide to Client/Server Architecture’, Client/Server Architecture, pp. 02-09.

6 Fahd K, University (2006), ‘Software Design and Architecture’, Client/Server Architecture, pp. 04-09.

7 Google, ‘Google, Images 2014’. Available at: ‘http://bit.ly/1qshnnZ’. Accessed time: Wednesday, October 08, 2014, 10:09pm.