International Journal of Advanced Science and Computer Applications (2024); 3(2):
DOI: 10.47679/ijasca.v3i1.48
https://ijasca.org/index.php/ijasca

RESEARCH ARTICLE

Client-server Architecture, a Review

Geofrey Nyabuto

Published online: 8 January 2024

Abstract

Client-server architecture is a software model through which resources and requests are serviced over a network. The client requests a resource over a network, and the server receives the request, processes it, and responds appropriately. With this model, multiple users can simultaneously access and use resources. This paper provides an overview of the architecture, outlining its characteristics, advantages, disadvantages, different implementations of the architecture as well as the current and future of this architecture.

Keyword: Client-server architecture, two-tier, three-tier, n-tier, cloud computing, microservices, inter-process communication

Introduction

The client-server architecture describes a computing model where one computer called a client requests a resource from another computer called a server over a network connection. The server receives the request, processes it, and responds to the client. In this model, there can be one or more client computers that request resources or services from one or more servers working together to service the request. The server usually has a database where it stores its data and runs programs that enable it to receive and process requests (Sharanagowda, 2022). Standardized protocols exist that enable the client and server to communicate. They include hypertext transfer protocol (HTTP), file transfer protocol (FTP) and simple mail transfer protocol (SMTP). This architecture provides inter-process communication between the client and the server through which they can exchange data (Kratky & Reichenberger, 2013). Many applications are running on this model including email exchange, database systems and the Internet.

Requests emanate from the client, and it is sent to the server through a communication channel like a network. Clients could be simple computers that run client programs like web browsers, mobile applications or any other application that can request a service or resource. Such computers do not need many configurations or complex programs since they do not service any request. On the other hand, servers are complex computers with high processing power based on the programs they are running on. The server receives typical requests, processes them, and responds to the client (Kumar, 2019). Constructing a server needs complex privileges as they need to run many programs including security mechanisms that validate and authenticate user requests before allowing them to be processed.

Characteristics of Client-Server Architecture

1. Both the client and server computers need protocols through which they can communicate. They communicate directly with the transport layer protocol. The transport layer uses the lower layer protocols to send and receive messages.
2. One server can service several requests simultaneously. However, each service will need a separate program to serve the requests.
3. The architecture has both horizontal and vertical scalability. More servers can be plugged into the architecture to serve the increasing workload. At the same time, server capabilities can be increased like RAM and CPU.
4. In the client-server architecture, client and server computers can run on heterogeneous hardware and software resources.

Advantages

1. In this architecture, all data is saved in a central location and accessed from multiple locations.
2. There is increased scalability and efficiency as more and more servers can be plugged into the architecture.
3. It is cheap to maintain resources in this architecture since they can be managed centrally.
4. It allows load balancing as the model allows plugging in redundant/replica servers that spread the workload across and at the same time offers an easier recovery option in case one server fails.
5. Makes resource sharing possible from different platforms.

Disadvantages

Kibabii University

*) corresponding author

Email: gmnyabuto@gmail.com

1. Security is a major challenge with this architecture since all information is stored centrally hence if the server or network is attacked, services may not be available. Attacks like denial-of-service attachment (DOS) will render the resources inaccessible. Man in the middle is possible as well as packet spoofing.
2. Initial implementation of this architecture is expensive as one needs to buy servers and implement networking and security configurations including firewalls.

Types of Client-Server Architecture

There exist different client-server architectures based on the number of servers involved in the implementation. Some of the common architectures include.

(1). Two-Tier Architecture

In this model, there are only two tiers of computing devices involved i.e., client and server. Workload is divided among these computers where the client will host user interfaces and be used to send requests to the server whereas the server will host the system that will be processing user requests. Such an architecture can be used in managing simple applications which do not need a lot of processing power.

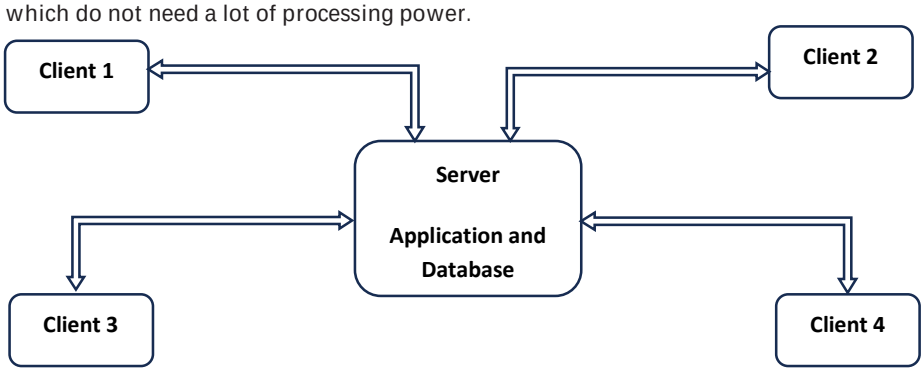

Fig 1. Two-tier Architecture

Advantages

1. High performance as the database and application are hosted in the system i.e., physically close to one another hence can quickly share information and service requests.
2. Developing applications that run on this architecture is easier as they don't need any extra logic for linking with other servers.
3. Applications developed in this environment are homogeneous as they have static business logic.

Disadvantages

The two-tier architecture is an excellent design for systems with very few users who concurrently access the system. However, as the number of users concurrently accessing the system increases, this model proves ineffective with the below challenges.

1. Performance becomes a challenge as the number of users accessing and using the system resources increases. This negatively impacts the implementation of such a system rendering it unusable. Extra mechanisms need to be put in place to ensure user requests are serviced.
2. The two-tier architecture applications face a problem of portability as the systems are tightly coupled. They are dependent on some databases, and migrating from one to another usually proves problematic. For example, porting a system running on MySQL to another RDBMS like MSSQL is usually very challenging.

(2). Three-Tier Architecture

This architecture introduces an additional tier to the two-tier architecture to help overcome challenges encountered while implementing it. In this model, the business logic, that is, the access of information, data storage and user interfaces are all hosted differently. The additional tier hosts the application system while the other hosts the database. This means the application system sits on a different server as well and the database or storage system sits on another server.

The three components of this architecture are.

Client Tier: This is the tier that displays information and exists at the highest level. It sends information to other tiers for processing and displays results by sending them to the browser.

Logic or Application Tier: This is the middle tier also called the logic or business logic layer. It does the processing of requests as received from the client tier, and fetches data from the data tier to help in processing and servicing requests.

Data Tier: In this tier, information is stored and retrieved from the database. This tier does the actual storage of data using a database management system. The application layer services requests by extracting necessary information from this tier.

Fig 2. Three-tier Architecture

Advantages

1. Easier maintainability: With the separation of concerns and modularization of systems, it is easier to maintain such a system.
2. Improved scalability: It is easier for each of these servers to scale up since they run separately.
3. Better security: Since the presentation, application and data layers are separated, there is improved security. Users are unable to directly manipulate the database as all requests must go through the application tier.
4. Increased performance as data and business logic are separated and each server or tier focuses on the processing of its core requests as required.

Disadvantages

1. Compared to a two-tier architecture, it is complex to build a three-tier due to the increased number of communication points.
2. This model may call for an implementation of proxy server that may end up increasing traffic.

(3). N-Tier Architecture

Also called multi-tier architecture, this model has presentation, processing and data management logic separated both logically and physically. In this architecture, several servers come together to process a request. The architecture has clearly defined or separate layers with each implementing a specific functionality.

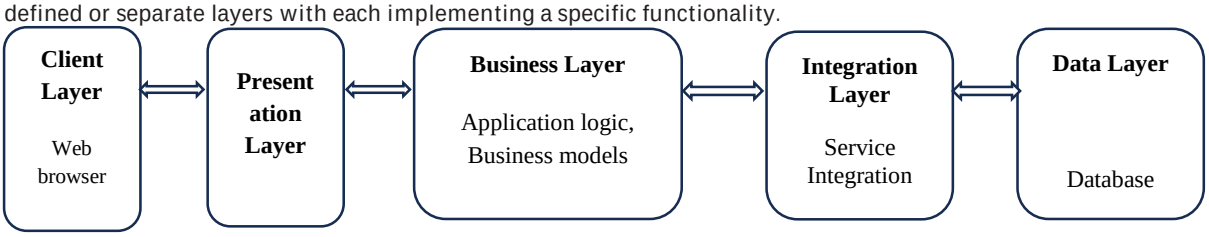

Fig 3. N-tier Architecture

This model has several advantages and applications. It has widely been applied in developing scaling applications like customer relationship management systems (CRM) as it has proved to be efficient, fast, and secure (Bhardwaj, Pandya, & Patel, 2014). N-tier architecture is scalable compared to the other architectures and helps improve data integrity. It is easier to create reusable components with improved security in this architecture. Though this architecture has many advantages, it is very complex to design applications in this architecture as a lot of effort is needed to program, deploy, and maintain these applications.

Result and Discussion

Recent and Future Development

Connected computing devices can share information through the Internet. The concurrent sharing of data by several devices has made it possible to have interventions like massively multiplayer online games where users can play and compete virtually (Ali, Alauldeen, & Khamees, 2020). The success of applications on the internet is made possible through the client-server architecture making it possible for different users from different locations to share and access information stored centrally.

Cloud Computing

With increased adoption and penetration of IT in every aspect of human life, more data will continuously be generated by different devices that will ultimately need storage. This ideally means there will be an increased need for data centers to store this data. Increased internet speeds will further push the use of cloud computing technologies. Technologies like IoT, AI, and ML, among others, will further push the demand for cloud computing services as they rely on the internet and data for their success and recent research already shows their increased adoption (Islam, et al., 2023).

Microservices

Due to the increased demand for high-performance software systems, there is a need to develop modular software systems that are based on the microservice architecture. Microservice architecture advocates for the development of independently deployed, loosely coupled software systems which leads to the rapid development of highly scalable and reliable software systems. The client-server architecture allows rapid deployment and communication between the different services (Zhang, Pang, Xu, & Niu, 2023).

Conclusion

Client-server architecture is one of the most popular models making use of both the client and server hardware resources to service a request. This architecture can be used in both the local network LAN and on the Internet. The server ensures all requests received from the client are serviced. Many applications, for example, mail systems, printing systems, ATM and others are based on this model.

Cloud computing, a buzzword in the modern computing era is based on this architecture as well as micro services. With this architecture, we can have simple implementations like 2-tier architecture or complex n-tier architecture that can run several services at the same time.

References

[1] Ali, S., Alauldeen, R., & Khamees, R. A. (2020). What is Client-Server System: Architecture, Issues and Challenge of Client -Server System (Review). *ResearchGate*, 1-6.

[2] Bhardwaj, D., Pandya, D., & Patel, D. (2014). Implementing N-Tier Architecture for Improvement in Customer Relationship Management "CRM". *International Journal of Engineering Research & Technology (IJERT)*, 2205-2209.

[3] Islam, R., Patamsetti, V. V., Gadhi, A., Gondu, R. M., Bandaru, C. M., Kesani, S. C., & Abiona, O. (2023). The Future of Cloud Computing: Benefits and Challenges. *International Journal of Communications, Network and System Sciences*, 53-65.

[4] Kratky, S., & Reichenberger, C. (2013). *Client/Server Development based on the Apple Event Object Model.* Atlanta. Retrieved from http://preserve.mactech.com/articles/mactech/Vol.14/14.11/Client-ServerDevelopment/index.html

[5] Kumar, S. (2019). A Review on Client-Server Based Applications And Research Opportunity. *International Journal of Recent Scientific Research*, 3857-33862.

[6] Sharanagowda, K. (2022). A Study on the Client Server Architecture and Its Usability. *IOSR Journal of Computer Engineering*, 73-76.

[7] Zhang, L., Pang, K., Xu, J., & Niu, B. (2023). High-performance microservice communication technology based on modified remote procedure call. *Scientific Reports*, 1-17.