Murang'a University of Technology

Innovation for Prosperity

DEPARTMENT Of INFORMATION TECHNOLOGY

Unit Code: SIT404

Title: CLIENT SERVER SYSTEMS

Course Notes

Lecturer's Name: J. Njuki

Purpose of the Course

Provide in-depth concepts of data communications, and networking and the concept of client server technology.

EXPECTED LEARNING OUTCOMES

At the end of this course the learner should be able to:

i) Design moderately complex networks using appropriate network and link types.

ii) Configure different types of network hardware.

iii) Physically construct different types of network cables.

Session Objectives

By the end of the session the student should be able to:

i) Explain the meaning of client/server systems

ii) Identify the components of a client server systems

iii) Highlight the characteristics of the client and the server

iv) Explain the advantages and disadvantages of client/server systems

Terminologies

Applications Programming Interface (API)

A set of function and call programs that allow clients and servers to intercommunicate

Client

A networked information requester, usually a PC or workstation, that can query database and/or other information from a server

Middleware

A set of drivers, APIs, or other software that improves connectivity between a client application and a server

Terminologies Cont'd

Relational Database

A database in which information access is limited to the selection of rows that satisfy all search criteria

Server

A computer, usually a high-powered workstation, a minicomputer, or a mainframe, that houses information for manipulation by networked clients

Structured Query Language (SQL)

A language developed by IBM and standardized by ANSI for addressing, creating, updating, or querying relational databases

DISTRIBUTED SYSTEMS ARCHITECTURE

A distributed system, also known as distributed computing, is a system with multiple components located on different machines that communicate and coordinate actions in order to appear as a single coherent system to the end-user.

The machines that are a part of a distributed system may be computers, physical servers, virtual machines, containers, or any other node that can connect to the network, have local memory, and communicate by passing messages

A distributed system is one in which both data and transaction processing are divided between one or more computers connected by a network, each computer playing a specific role in the system

HOW distributed systems function

There are two general ways that distributed systems function:

- Each machine works toward a common goal and the end-user views results as one cohesive unit.
- Each machine has its own end-user and the distributed system facilitates sharing resources or communication services.

Although distributed systems can sometimes be obscure, they usually have three primary characteristics:

=>all components run concurrently,

=>there is no global clock, and

=>all components fail independently of each other.

Benefits and challenges of distributed systems

There are three reasons that teams generally decide to implement distributed systems:

Horizontal Scalability —Since computing happens independently on each node, it is easy and generally inexpensive to add additional nodes and functionality as necessary.

Reliability —Most distributed systems are fault-tolerant as they can be made up of hundreds of nodes that work together. The system generally doesn't experience any disruptions if a single machine fails.

Performance —Distributed systems are extremely efficient because work loads can be broken up and sent to multiple machines.

Challenges of Distributed Systems

Three more challenges you may encounter include:

Scheduling — A distributed system has to decide which jobs need to run, when they should run, and where they should run. Schedulers ultimately have limitations, leading to underutilized hardware and unpredictable runtimes.

Latency —The more widely your system is distributed, the more latency you can experience with communications. This often leads to teams making tradeoffs between availability, consistency, and latency.

Observability —Gathering, processing, presenting, and monitoring hardware usage metrics for large clusters is a significant challenge

Types of distributed systems

Distributed systems generally fall into one of four different basic architecture models:

Client-server —Clients contact the server for data, then format it and display it to the end-user. The end-user can also make a change from the client-side and commit it back to the server to make it permanent.

Three-tier —Information about the client is stored in a middle tier rather than on the client to simplify application deployment. This architecture model is most common for web applications.

n-tier —Generally used when an application or server needs to forward requests to additional enterprise services on the network.

Peer-to-peer —There are no additional machines used to provide services or manage resources. Responsibilities are uniformly distributed among machines in the system, known as peers, which can serve as either client or server.

Introduction to client/ server systems

- We are in the midst of a fundamental change in both technology and its application.
- Organizations today expect to get more value from their investments in technology
- According to MIS terminology, Client/Server computing is new technology that yields solutions to many data management problems faced by modern organizations.

Client/server systems

By definition a client-server system is a computing system in distributed computing that is using a software engineering technique where it enables two or more processes that are not related to each other to exchange information in the common network by using a widely known protocol

Client/server systems cont'd

Client/server is a computing **system** that is composed of two logical parts:

- A **server**, which provides services, and
- A **client**, which requests them.

The two parts can run on separate machines on a network, allowing users to access powerful **server** resources from their personal computers.

Client/server systems cont'd

Client - A client is a single-user workstation that provides presentation services and the appropriate computing, connectivity and the database services and the interfaces relevant to the business need.

Server- A server is one or more multi-user processors with shared memory providing computing, connectivity and the database services and the interfaces relevant to the business need.

Client/server systems cont'd

- The Client/Server computing is an environment that satisfies the business need by appropriately allocating the application processing between the client and the server processors.
- The protocol is the client requests the services from the server; the server processes the request and returns the result to the client.
- The communication mechanism is a message passing Inter-Process communication (IPC) that enables the distributed placement of the client and server processes

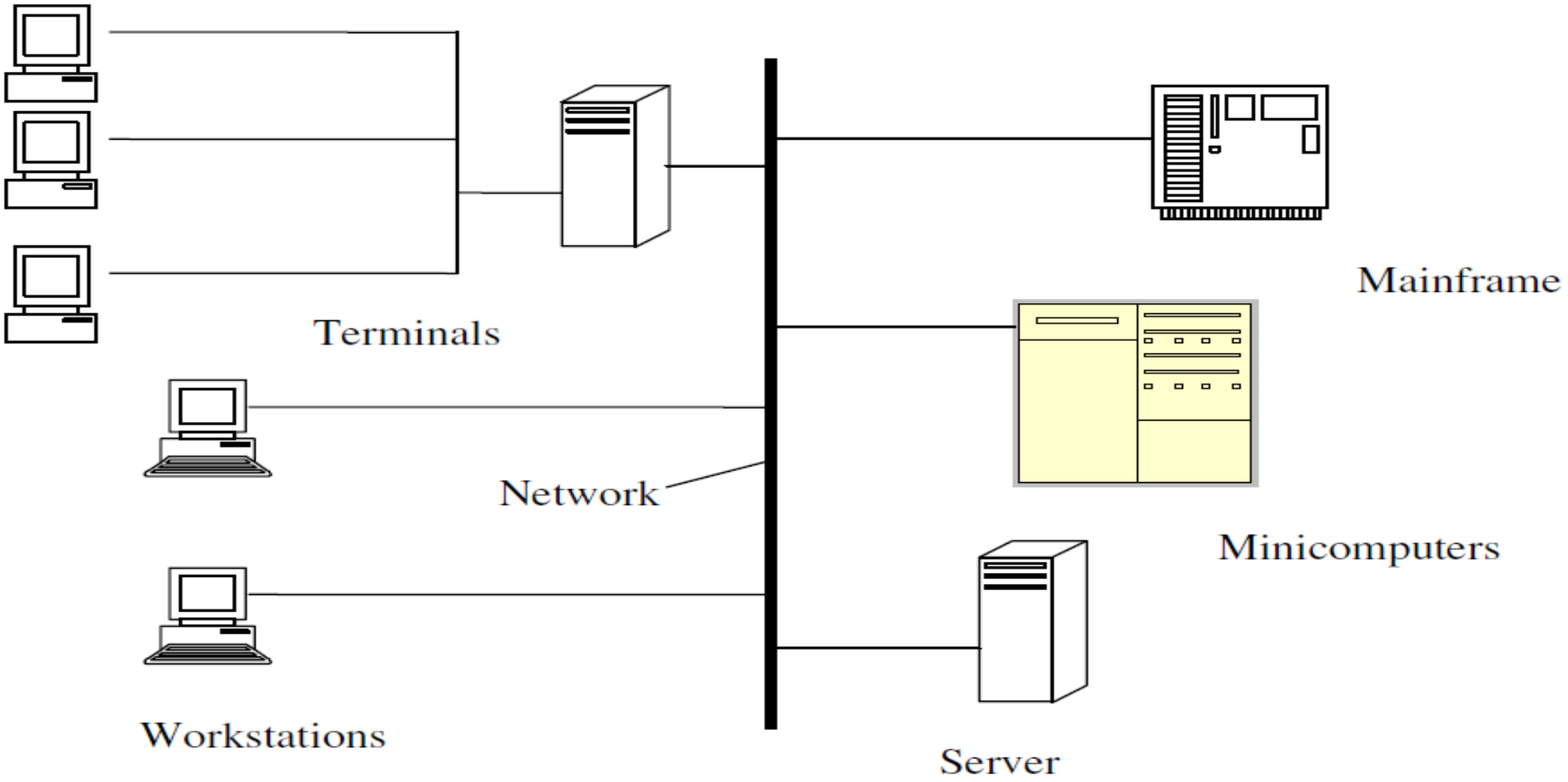
Client/server systems cont'd

- The Client/Server is the generic model and fits what is know in the industry as the "cooperating processing" or "peer-to-peer".
- The client/server computing is fundamentally platform independent.
- The user of an application wants the functionality (business) it provides; the computing platform provides access to this business functionality.
- There are no benefits but a considerable amount of risk of exposing the platform to the users.

Client/server systems cont'd

- The changes in the platform and the underlying technology should be transparent to the user, as it is understood that the systems built with transparency to the technology, for the entire user offer the highest probability of solid ongoing return for the technology investment.
- It is also easily demonstrable that the developers become aware of the target platform, development will be bound to that platform and they will use special tricks and feature found in that specific platform.

Modern Client/Server Architecture



Components of client-server

Three components that are essential in client-server architecture

i) Client

A client, also known as workstation, operates the front-end applications.

ii) Server

A server, usually known as the back-end, is a machine that manages the roles commanded for concurrent, shared data access, and also can be referred as a server process that operates on a server machine.

iii) Network

A network allows distant data retrieval via client-server and server-to-server transmission.

Characteristics of the Client and the Server

- The clients and the servers are the logical entities that work together over a network to accomplish a task

1) Service: The client/server is primarily a relationship between processes running on separate machines. The server process is a provider of services. The client is a consumer of services. In essence, client/server provides a clean separation of function based on the idea of service.

2) Shared Resources: A server can service many clients at the same time and regulate their access to shared resources

Characteristics of the Client and the Server cont'd

3) Asymmetrical protocols: There is a many-to-one relationship between the clients and the server. Clients always initiate the dialog by requesting a service.

4) Transparency of location: The server is a process that can reside on the same machine as the client or on a different machine across a network. Client/Server software usually masks the location of the server from the clients by the redirecting the service calls when needed. A program can be a client, a server, or both.

5) Mix-and-match: The ideal client/server software is independent of hardware or operating system software platforms. You should be able to mix-and-match client and server platforms.

Characteristics of the Client and the Server cont'd

6) Message-based exchanges: Clients and servers are loosely coupled systems that interact through a message-passing mechanism. The message is the delivery mechanism for the service request and replies.

7) Encapsulation of services: The server is a specialist. A message tells a server is requested; it is then up to the server to determine how to get the job done. Servers can be upgraded without affecting the clients as long as the published message interface is not changed.

Characteristics of the Client and the Server cont'd

8) Scalability: Client/Server systems can be scaled horizontally or vertically. Horizontal scaling means adding or removing client workstations with only a slight performance impact. Vertical scaling means either migrating to a larger and faster server machine or distributing the processing load across multiple servers.

9) Integrity: The server code and server data is centrally managed, which results in cheaper maintenance and the guarding of shared data integrity. At the same time, the clients remain personal and independent.

Advantages of client/server systems

1) Enhanced Data Sharing

Data that is collected as part of the normal business process and maintained on a server is immediately available to all authorized users.

2) Integrated Services

In the client/server model, all information that the client is entitled to use is available at the desktop.

3) Sharing Resources among Diverse Platforms

The client/server computing model provides opportunities to achieve true open system computing. Applications may be created and implemented without regard to the hardware platforms or the technical characteristics of the software

Advantages of client/server systems cont'd

4) Data Interchangeability and Interoperability

SQL is an industry-standard data definition and access language. This standard definition has enabled many vendors to develop production-class database engines to manage data as SQL tables.

5) Location Independence of Data and Processing

Users log into an application from the desktop with no concern for the location or technology of the processors involved.

6) Client-server architecture enables the roles and responsibilities of a computing system to be distributed among several independent computers that are known to each other only through a network

Advantages of client/server systems cont'd

7) All the data is stored on the servers, which generally have far greater security controls than most clients. Servers can better control access and resources, to guarantee that only those clients with the appropriate permissions may access and change data. Since data storage is centralized, updates to that data are far easier to administer than what would be possible

8) Many mature client-server technologies are already available which were designed to ensure security, 'friendliness' of the user interface, and ease of use. It functions with multiple different clients of different capabilities.

Disadvantages of client/server systems

Comparisons with the P2P

1. Traffic congestion on the network has been an issue. As the number of simultaneous client requests to a given server increases, the server can become severely overloaded

Contrast that to a P2P network, where its bandwidth actually increases as more nodes are added, since the P2P network's overall bandwidth can be roughly computed as the sum of the bandwidths of every node in that network.

2. The client-server paradigm lacks the robustness of a good P2P network. Under client-server, should a critical server fail, clients' requests cannot be fulfilled

In P2P networks, resources are usually distributed among many nodes. Even if one or more nodes depart and abandon a downloading file, for example, the remaining nodes should still have the data needed to complete the download

Client/Server Computing: Client

Client machines are generally single-user workstations providing a user-friendly interface to the end user

=> client-based stations generally present the type of graphical interface that is most comfortable to users, including the use of windows and a mouse

=> Microsoft Windows and Macintosh OS provide examples of such interfaces

=> client-based applications are tailored for ease of use and include such familiar tools as the spreadsheet

Client/Server Computing: Server

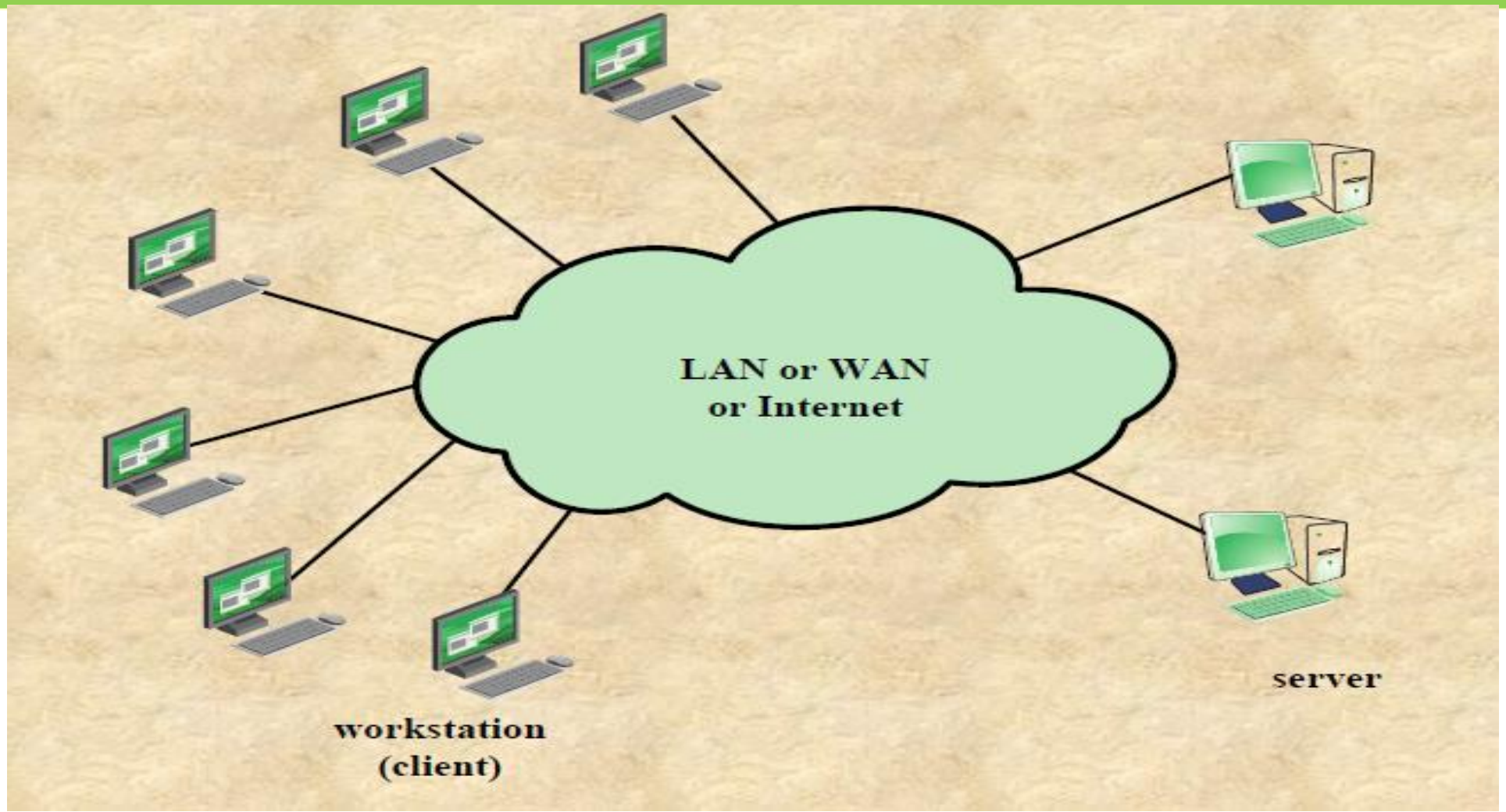
Each server provides a set of shared services to the clients

=> most common type of server currently is the database server, usually controlling a relational database

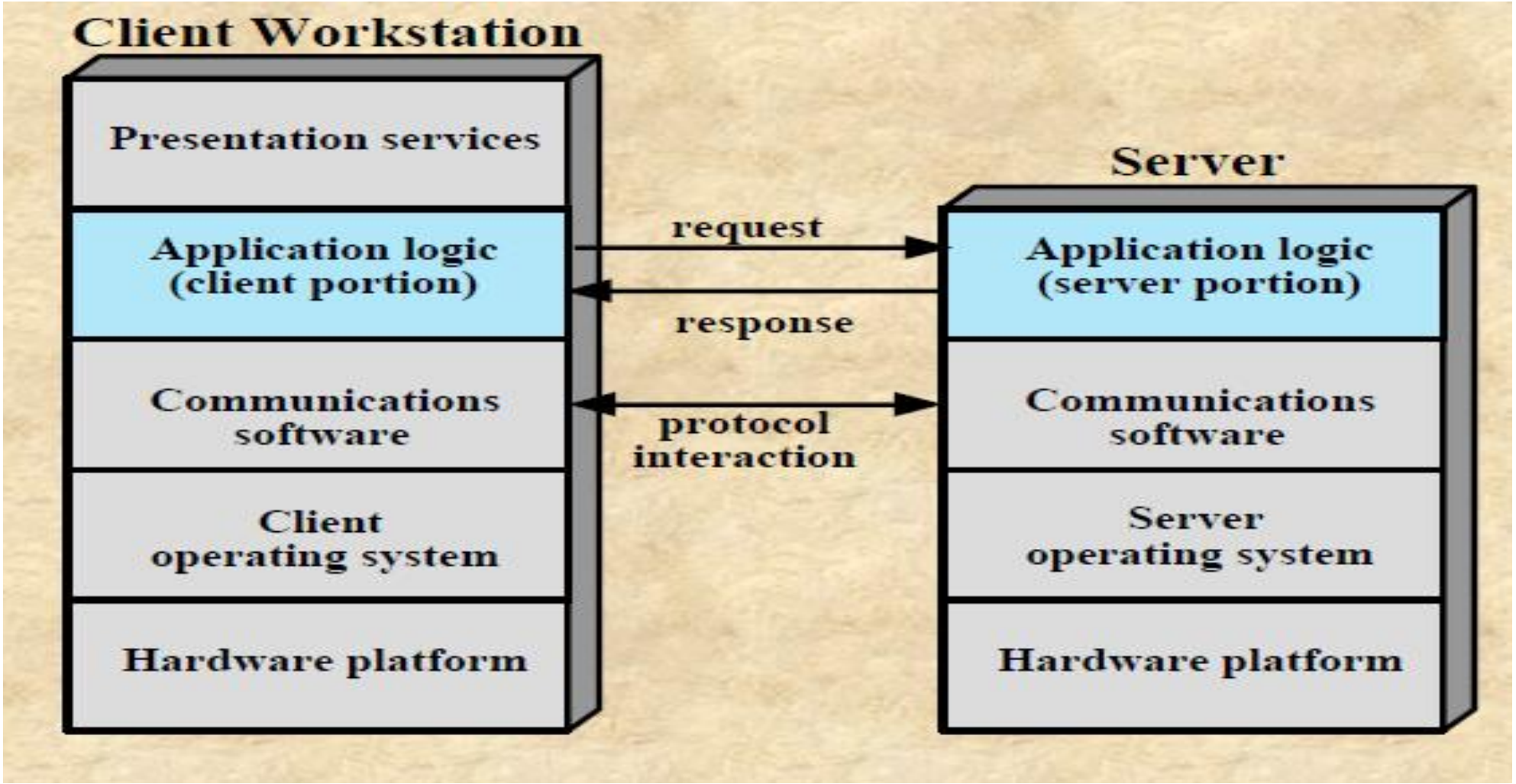
=> enables many clients to share access to the same database

=> enables the use of a high-performance computer system to manage the database

Generic Client/Server Environment



Generic Client/Server Architecture



Client/Server Applications

The key feature of a client/server architecture is the allocation of application-level tasks between clients and servers

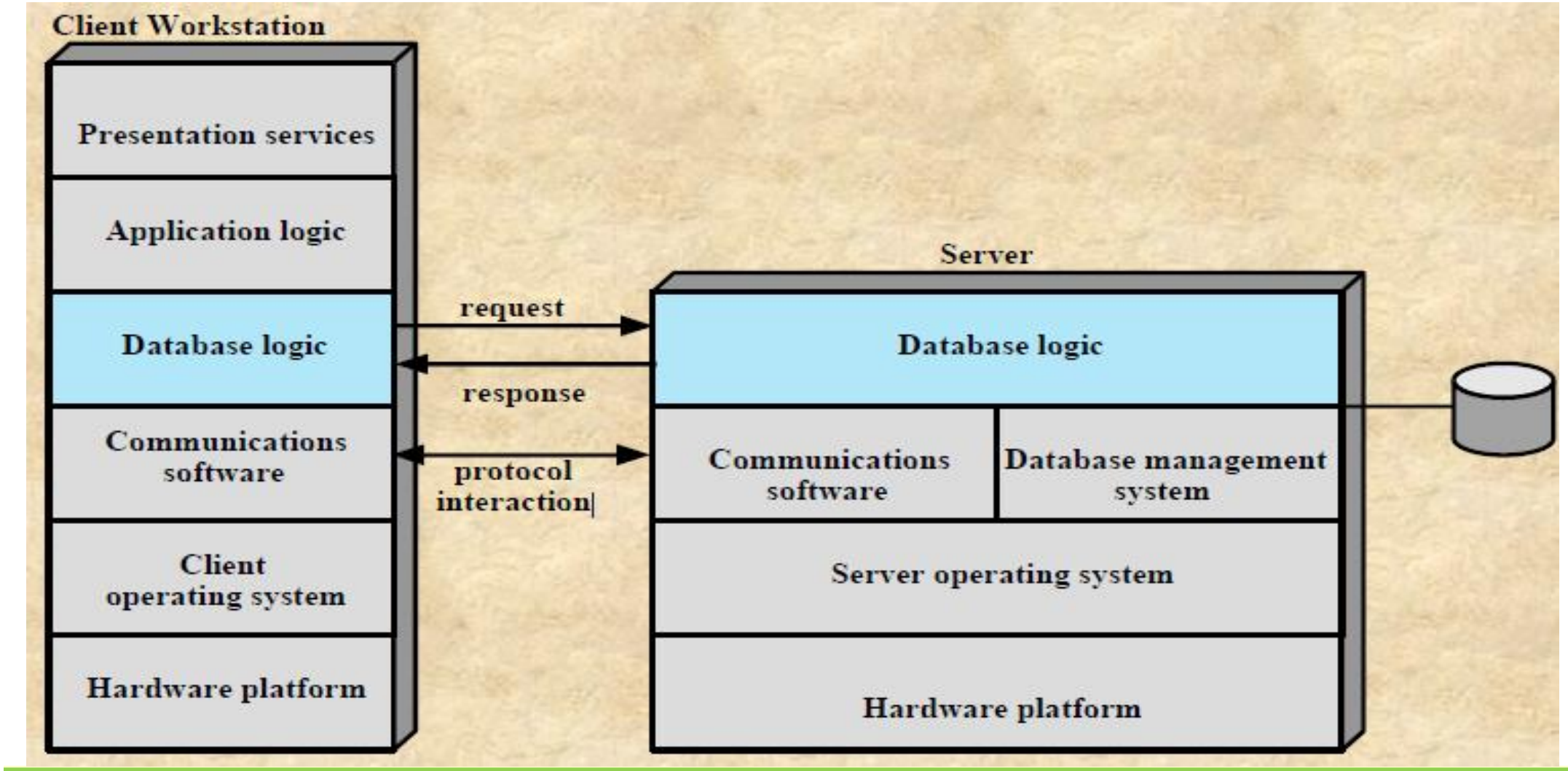
Hardware and the operating systems of client and server may differ

These lower-level differences are irrelevant as long as a client and server share the same communications protocols and support the same applications

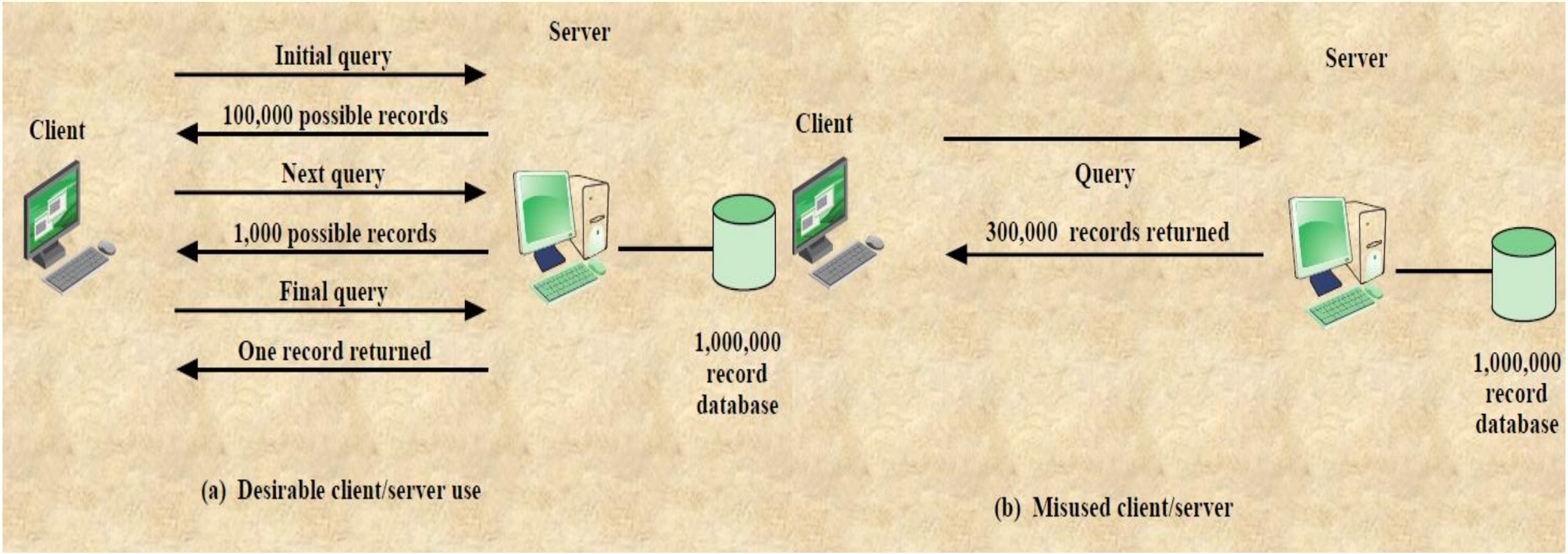
Client/Server Applications cont'd

It is the communications software that enables client and server to interoperate => principal example is TCP/IP
Actual functions performed by the application can be split up between client and server in a way that optimizes the use of resources
The design of the user interface on the client machine is critical => there is heavy emphasis on providing a graphical user interface (GUI) that is easy to use, easy to learn, yet powerful and flexible

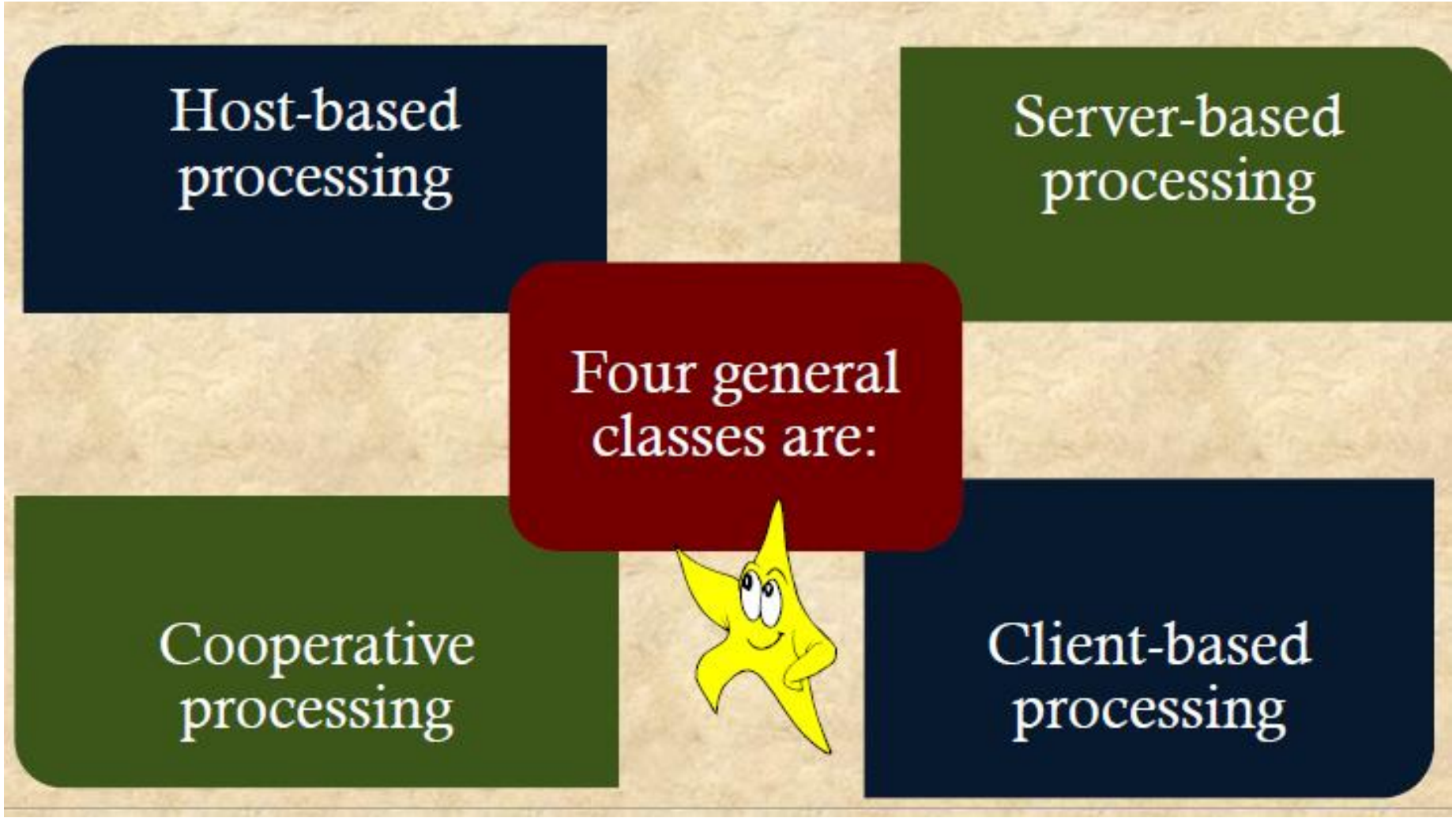
Client/Server Architecture for Database Applications



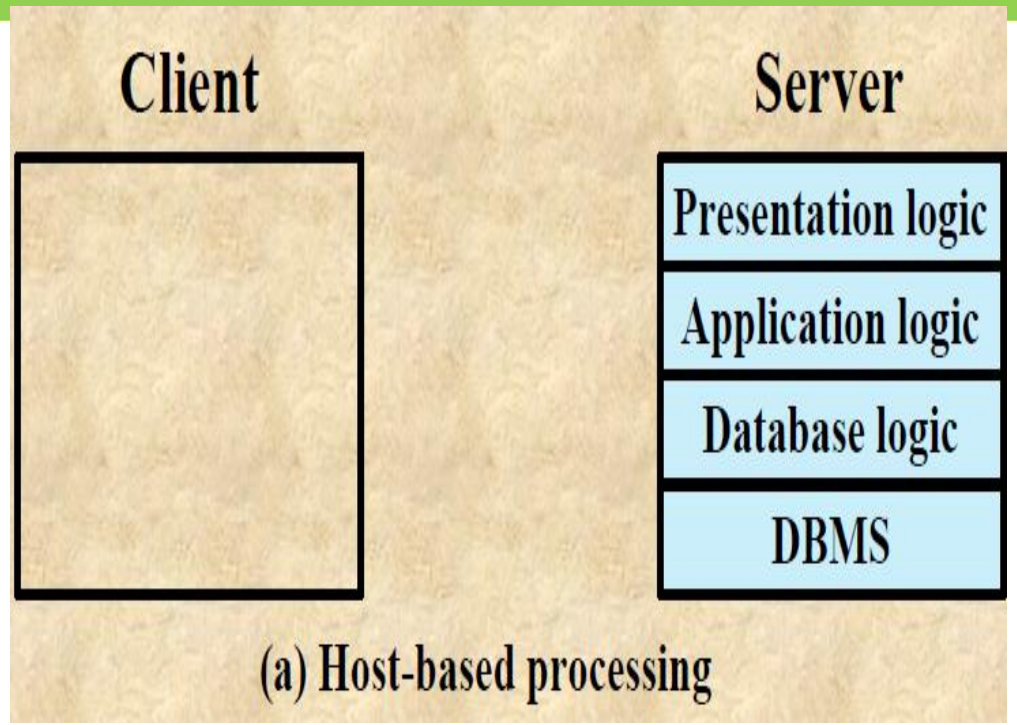
Client/Server Database Usage



Classes of Client/Server Applications

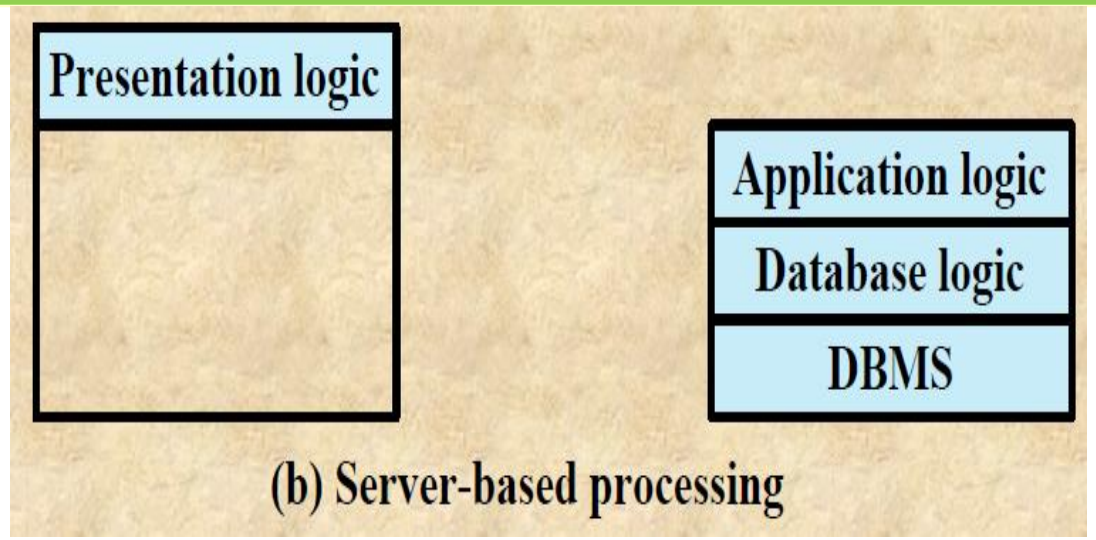


Classes of Client/Server Applications cont'd



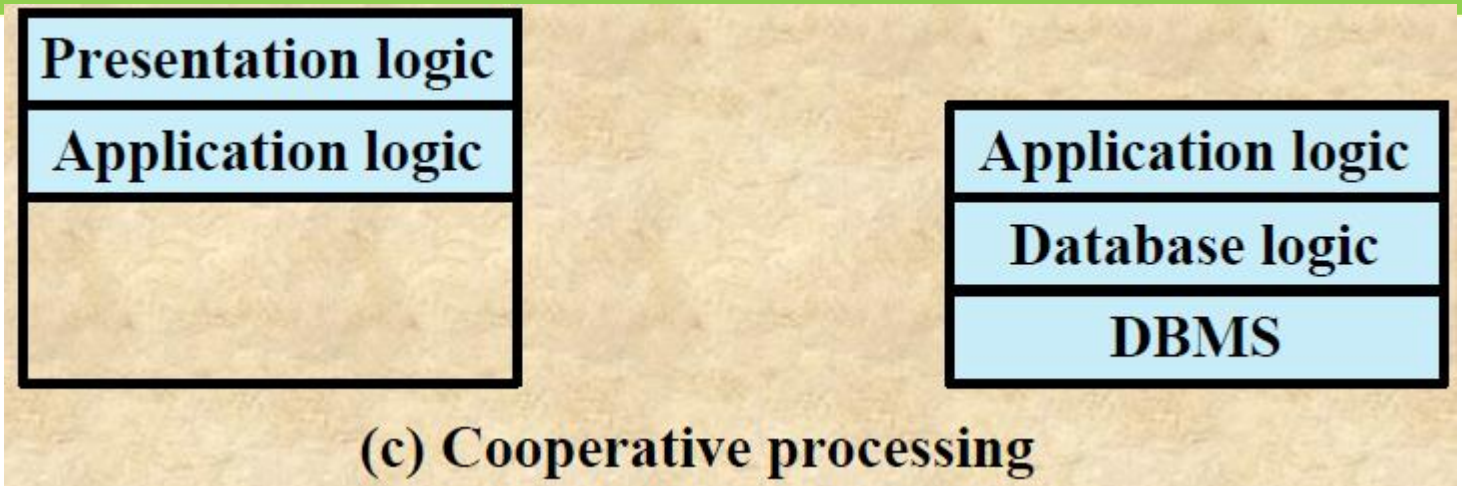
Not true client/server computing
=>Traditional mainframe environment in which all or virtually all of the processing is done on a central host
=> Often the user interface is via a dumb terminal
=>The user's station is generally limited to the role of a terminal emulator

Classes of Client/Server Applications cont'd



=>Server does all the processing
=>Client provides a graphical user interface
=>Rationale behind configuration is that the user workstation is best suited to providing a user-friendly interface and that databases and applications can easily be maintained on central systems
=> User gains the advantage of a better interface

CLASSES OF CLIENT/SERVER APPLICATIONS CONT'D

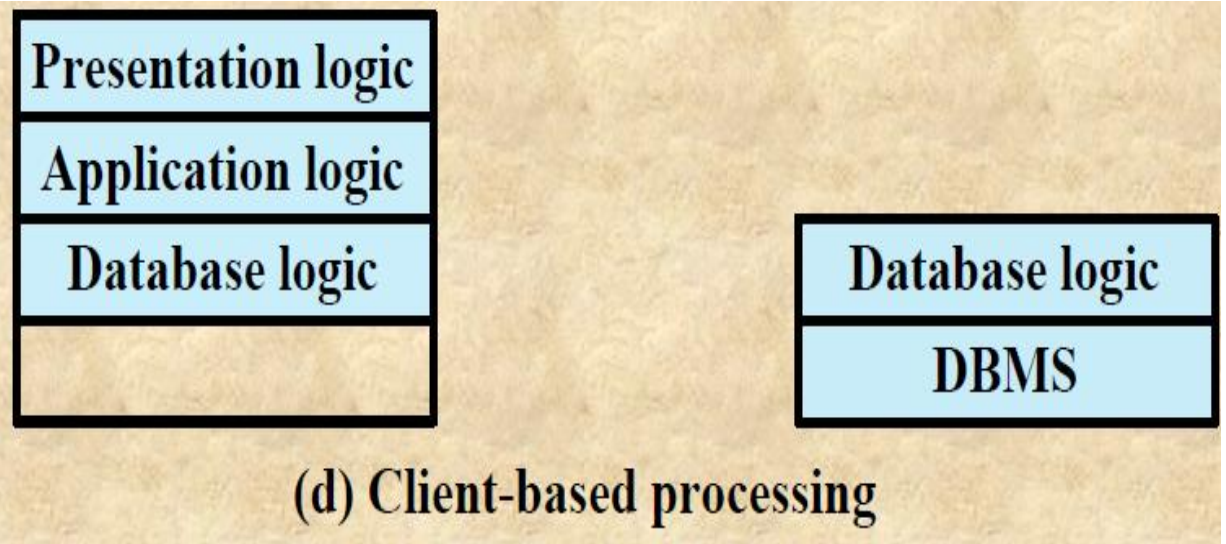


=> Application processing is performed in an optimized fashion

=> Complex to set up and maintain

=> Offers greater productivity and efficiency

Classes of Client/Server Applications cont'd



=> All application processing is done at the client

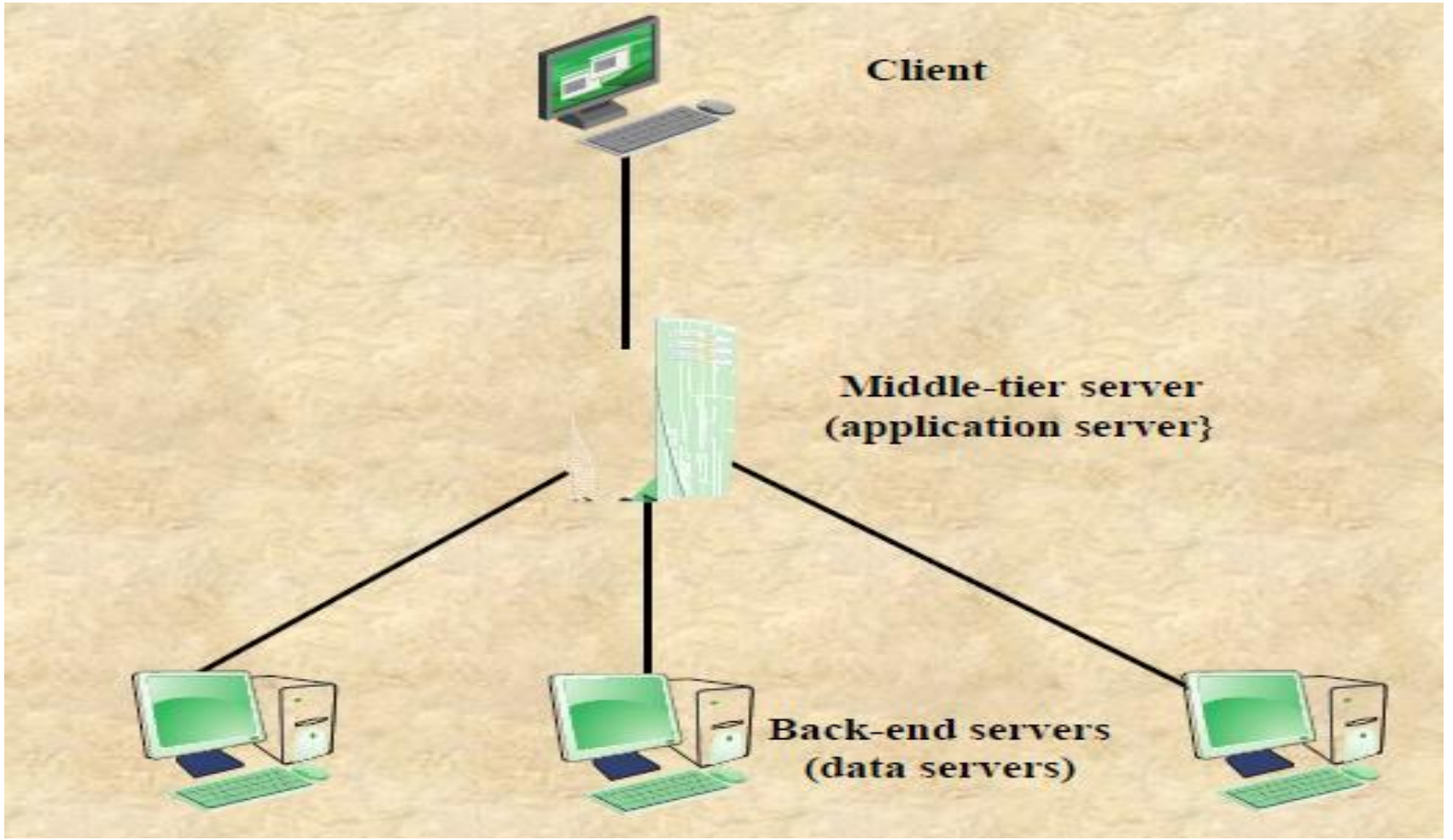
=> Data validation routines and other database logic functions are done at the server

=> Some of the more sophisticated database logic functions are housed on the client side

=> This architecture is perhaps the most common client/server approach in current use

=>It enables the user to employ applications tailored to local needs

Three-tier Client/Server Architecture



File Cache Consistency

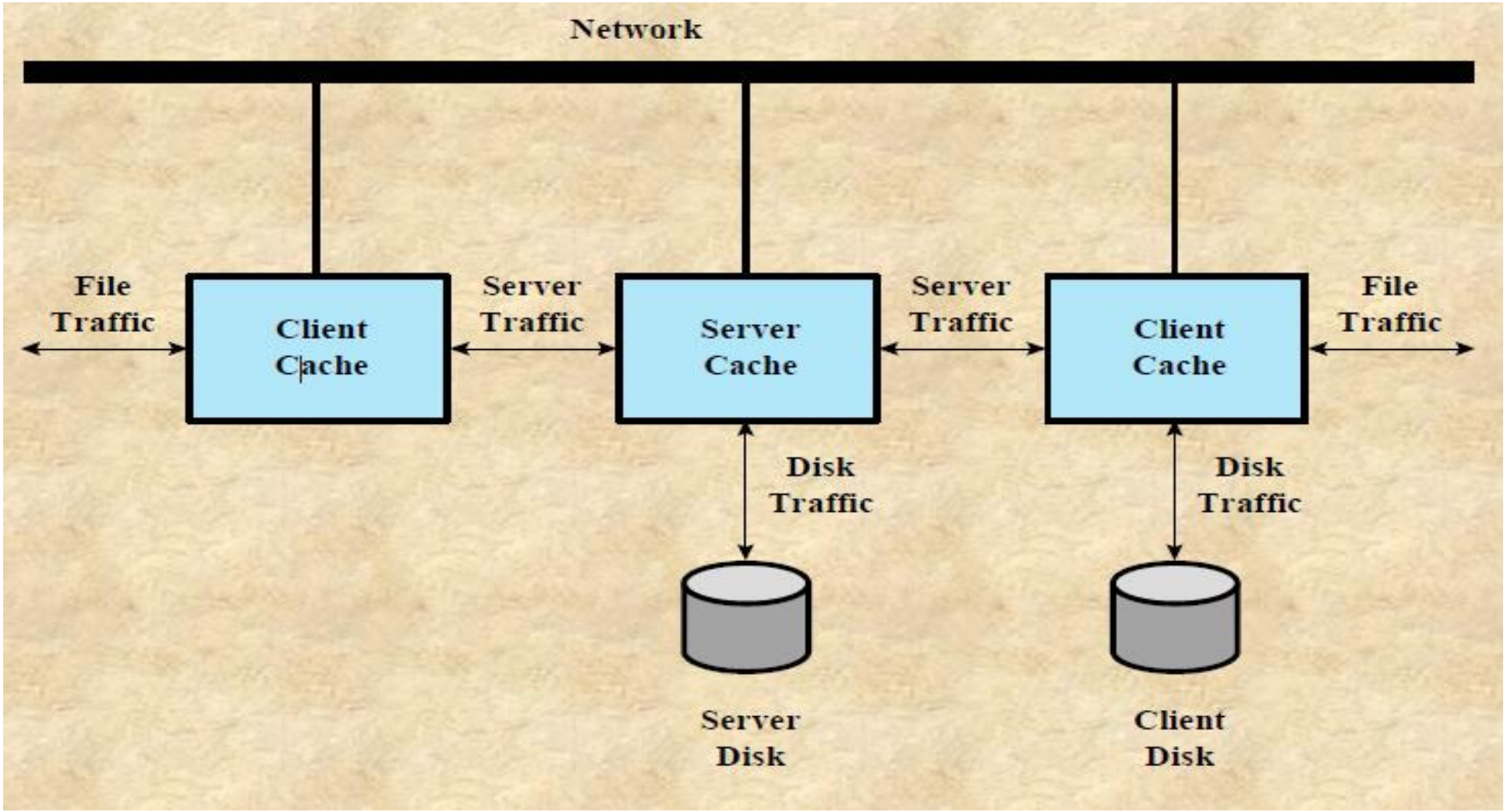
When a file server is used, performance of file I/O can be noticeably degraded relative to local file access because of the delays imposed by the network

=> File caches hold recently accessed file records

=> Because of the principle of locality, use of a local file cache should reduce the number of remote server accesses that must be made

The simplest approach to cache consistency is to use file locking techniques to prevent simultaneous access to a file by more than one client

Distributed File Caching



Middleware

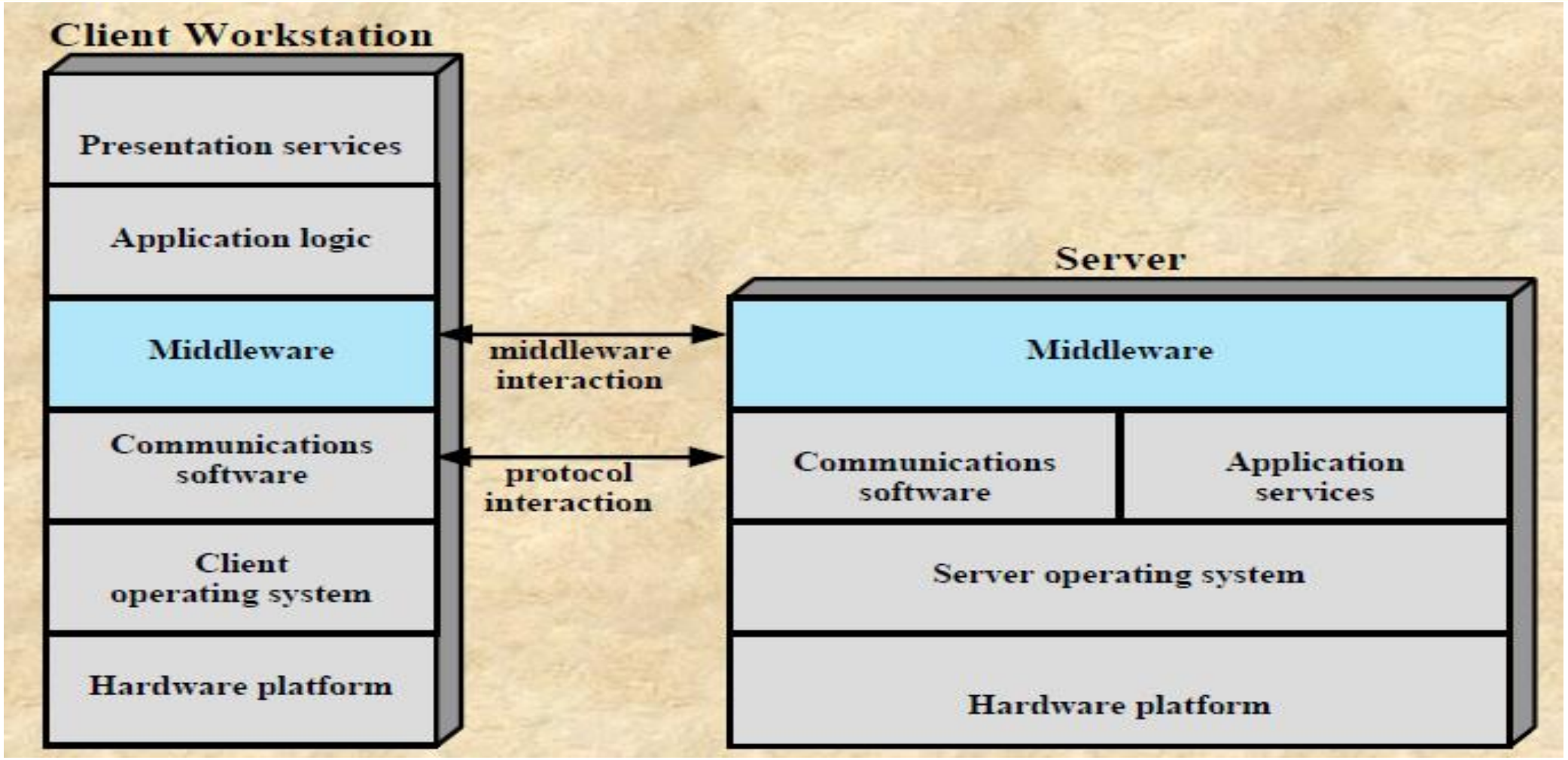
To achieve the true benefits of the client/server approach developers must have a set of tools that provide a uniform means and style of access to system resources across all platforms

=>This would enable programmers to build applications that look and feel the same

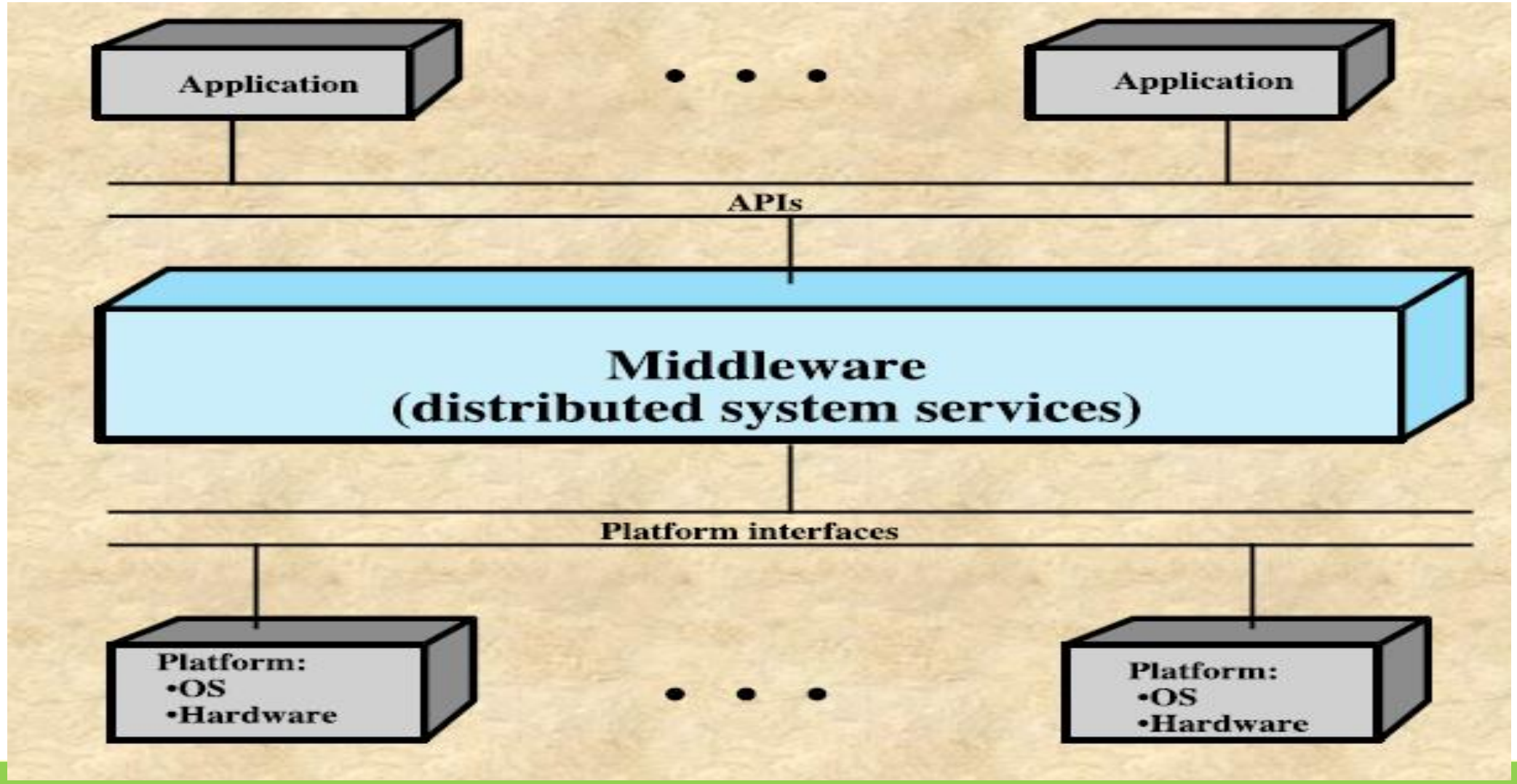
=>Enable programmers to use the same method to access data regardless of the location of that data

=>The way to meet this requirement is by the use of standard programming interfaces and protocols that sit between the application (above) and communications software and operating system (below)

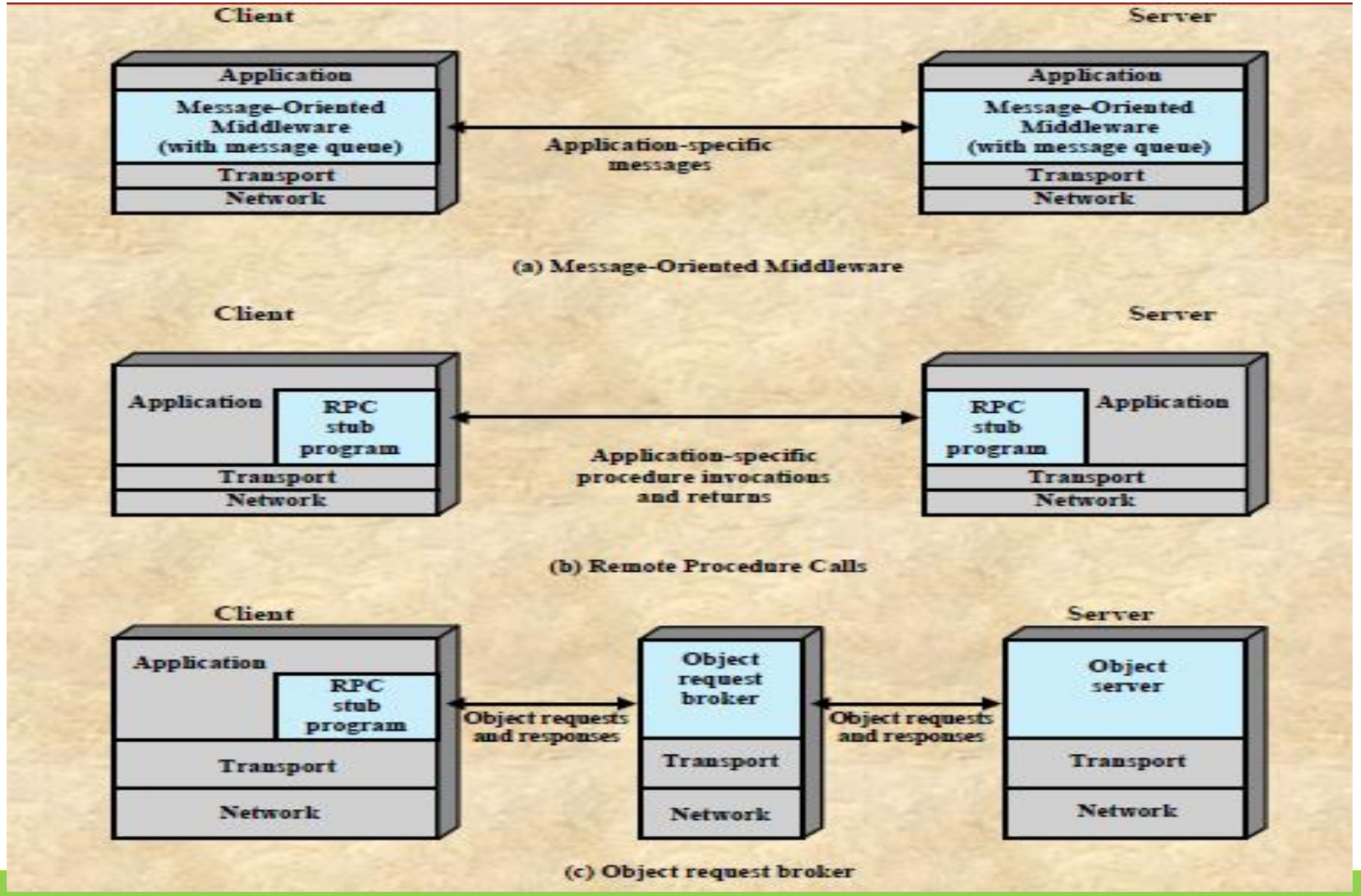
The Role of Middleware in Client/Server Architecture



Logical view of middleware



Middleware Mechanism



Summary

We have explained the meaning of:

- ⇒Distributed computing systems
- ⇒Client/server systems

Identified types of distributed systems:

- ⇒Client/server
- ⇒Peer to peer
- ⇒Three tier
- ⇒N-tier

We have highlighted the components of client/server systems

- ⇒Client
- ⇒Server
- ⇒Network

Explained characteristics, advantages and disadvantages of client/server systems


Q&A