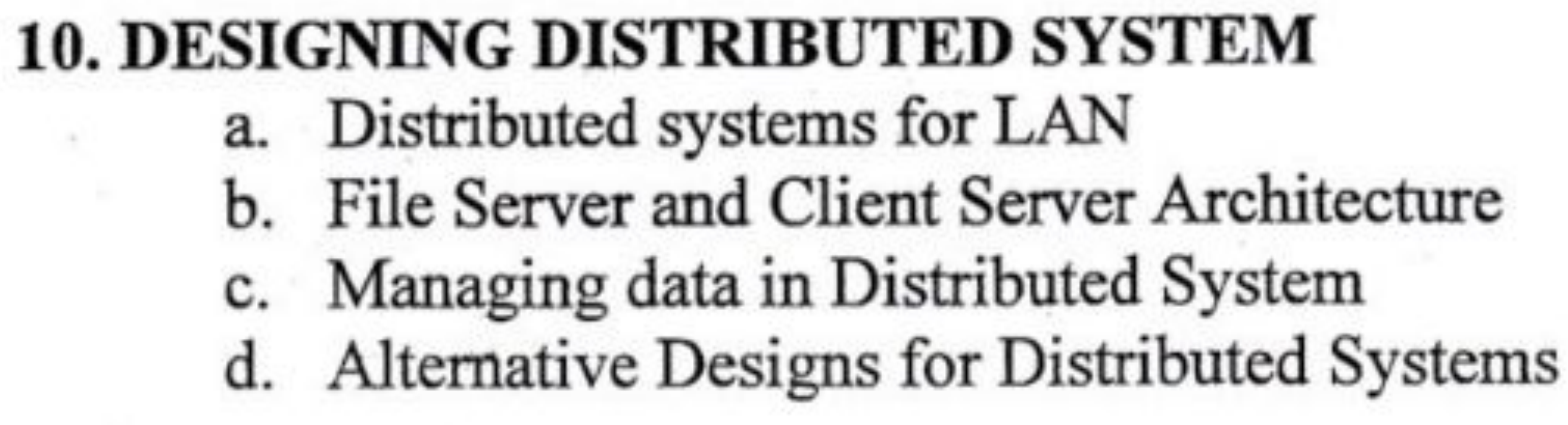
Chapter 10

Designing Distributed Systems

(3 Hours)

Chapter contents:



10. DESIGNING DISTRIBUTED SYSTEM

a. Distributed systems for LAN
b. File Server and Client Server Architecture
c. Managing data in Distributed System
d. Alternative Designs for Distributed Systems

What is a distributed system

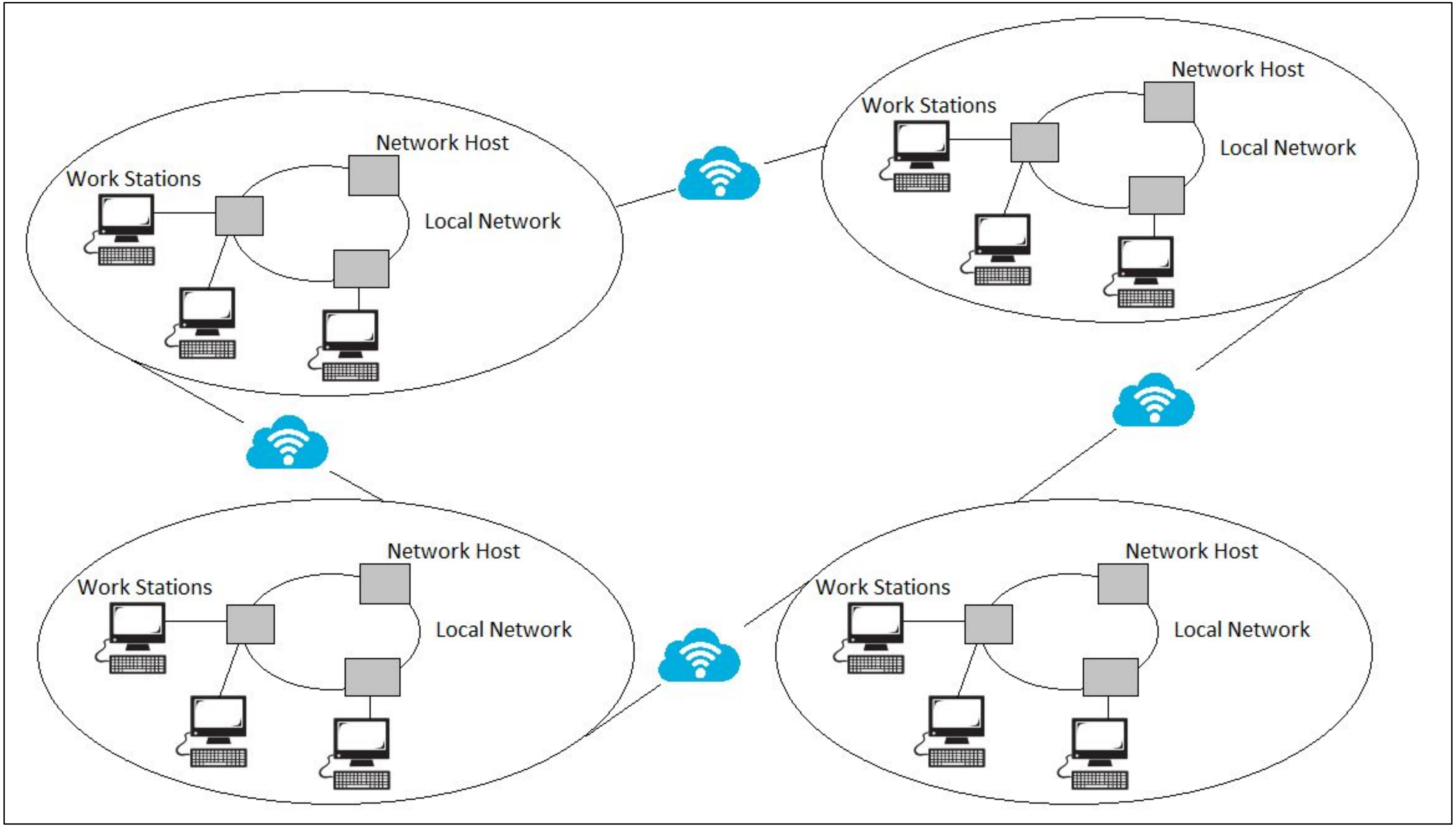
- a system with multiple components located on different machines that communicate and coordinate actions in order to appear as a single coherent system to the end-user.

- A collection of computers that are used jointly to perform a single operation or to provide a single service.

- 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.

Example of distributed systems

- Collection of Web Servers - more precisely, servers implementing the HTTP protocol— that jointly provide the distributed database of hypertext and multimedia documents that we know as the World-Wide Web
- DNS servers
- ATM networks etc.


Network Host
Work Stations
Local Network
Network Host
Work Stations
Local Network
Network Host
Work Stations
Local Network
Network Host
Work Stations
Local Network

Types of distributed systems

- **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.

Why Distributed System?

1. **Cost**. Better price/performance as long as commodity hardware is used for the component computers.

2. **Reliability**. By having redundant components the impact of hardware and software faults on users can be reduced.

3. **Inherent distribution**. Naturally and physically distributed

4. **Transparency**. The end users can be concealed or hidden from the actual separation of the distributed system so that the user feels that everything is transparent to everyone.

5. Scalability. Resources such as processing and storage capacity can be increased significantly.

6. Dependability. The dependence of a system on another system can be achieved to solve a particular task jointly.

7. Performance. By using the combined processing and storage capacity of many nodes, performance levels can be reached that are beyond the range of centralized machines.

8. Flexibility. Easily can be added or removed a node

Advantages of Distributed Systems over Centralized Systems

- **Economics**: a collection of microprocessors offer a better pp pp rice/performance than mainframes. Low price/performance ratio: cost effective way to increase computing power.

- **Speed**: a distributed system may have more total computing power than a mainframe. Ex. 10,000 CPU chips, each running at 50 MIPS. Not possible to build 500 000 MIPS single processor since it would require 0 002 nsec build 500,000 MIPS single processor since it would require 0.002 nsec instruction cycle. Enhanced performance through load distributing.

- **Inherent distribution**: Some applications are inherently distributed. Ex. a supermarket chain.

- **Reliability**: If one machine crashes, the system as a whole can still survive. Higher availability and improved reliability.

- **Incremental growth**: Computing power can be added in small increments. Modular expandability

- **Another deriving force**: the existence of large number of personal computers, the need for people to collaborate and share information.

Disadvantages of Distributed Systems

- Difficulty in developing software for distributed systems
- lossy transmissions
- Expense of maintenance such network
- Databases connected to the distributed systems are complicated and difficult to handle as compared to a single user system
- Chances of overloading in network if all the nodes try to communicate at once
- Easy access also applies to secret data!

The Process of Designing Distributed Systems

- Similar to designing single-location systems
- Due to multi-location deployment, numerous design issues must be considered
- More opportunity for failure due to number of components
- Understanding characteristics of commonly used architectures is key to successful design

The Process of Designing Distributed Systems

- Deliverables and Outcome
    - Document that consolidates system design information
        - Description of each site
        - Description of data usage for each site
        - Description of business process for each site
        - Contrasts of alternative IS architectures for site, data and processing needs of each site

Designing Distributed Systems

- Designing systems for local area networks (LAN)
    - LAN supports a network of personal computers that share common devices and software attached to the LAN
    - File Servers
        - A device that manages file operations
        - Shared by each client PC attached to the LAN
        - DBMS Applications
            - One copy of DBMS on file server and concurrently running copies on client PCs
            - All data manipulation is performed on the client PC

Designing Distributed Systems

- Designing systems for local area networks (LAN)
  - File Servers (continued)
    - Limitations
      - Excessive data movement
      - Need for powerful client workstation
      - Decentralized data control

Designing Distributed Systems

- Designing systems for a Client Server Architecture
  - Application processing is divided between client and server
  - Client manages the user interface
  - Database server is responsible for data storage and query processing
    - Database engine

Designing Distributed Systems

- Designing systems for a Client Server Architecture
  - Advantages
    - Allows companies to leverage the benefits of microcomputer technology
    - Allows processing to be performed close to the source of data
      - Improves response time
      - Reduces network traffic
    - Facilitates the use of graphical user interfaces (GUIs)
    - Encourages the acceptance of open systems
  - Disadvantages
    - Server must be more powerful than server in file server environment

Choosing Between File Server and Client/Server Architecture

- File server architecture
  - Supports only the distribution of data
  - Entire programs and databases must be transferred to clients when accessing system
  - Most appropriate for small applications with little or no concurrent data access

Choosing Between File Server and Client/Server Architecture

- Client/Server Architecture
  - Client and server share processing workload
  - Excellent choice for systems with large amounts of data and concurrent access needs

Advanced Forms of Client/Server Architecture

- Three-tiered client/server
  - Three logical and distinct applications
    - Data management
    - Presentation
    - Analysis
  - Middleware
    - Combination of hardware, software and communication technologies that bring together three distinct applications into one environment
  - Application Server
    - Server where data analysis functions are performed

Advanced Forms of Client/Server Architecture

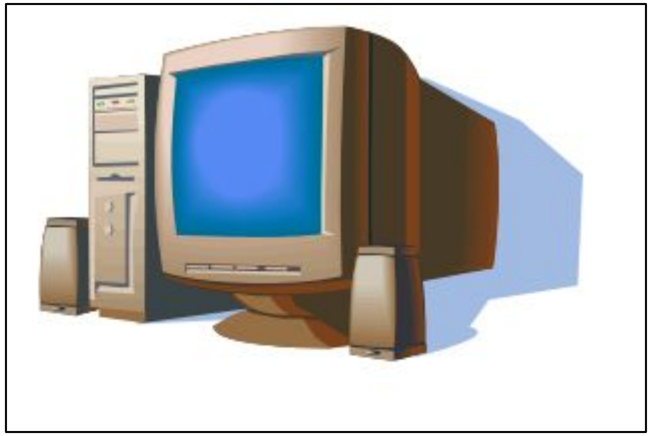
- Three-tiered client/server
  - Advantages
    - Applications can be partitioned in a way that best fits the organizational computing need
    - Easier customization
    - Easier maintenance

Client-Server Architectures in distributed Systems

1. Client Application model (1-tier)
2. Client server interaction model (2-tier)
3. Layered application model (3-tier)
4. Generic layered application model (n-tier)

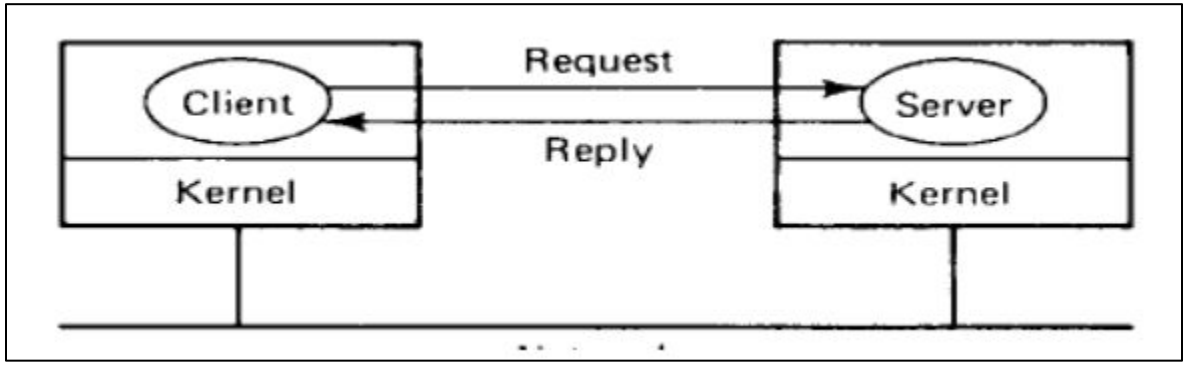
Client Application Model (1-tier)

- The program runs on a single machine.
- In most cases there is no separate application layers.
- Database applications are on the same machine.
- At any given moment, only one user can use the program



Client-server interaction model (2-tier)

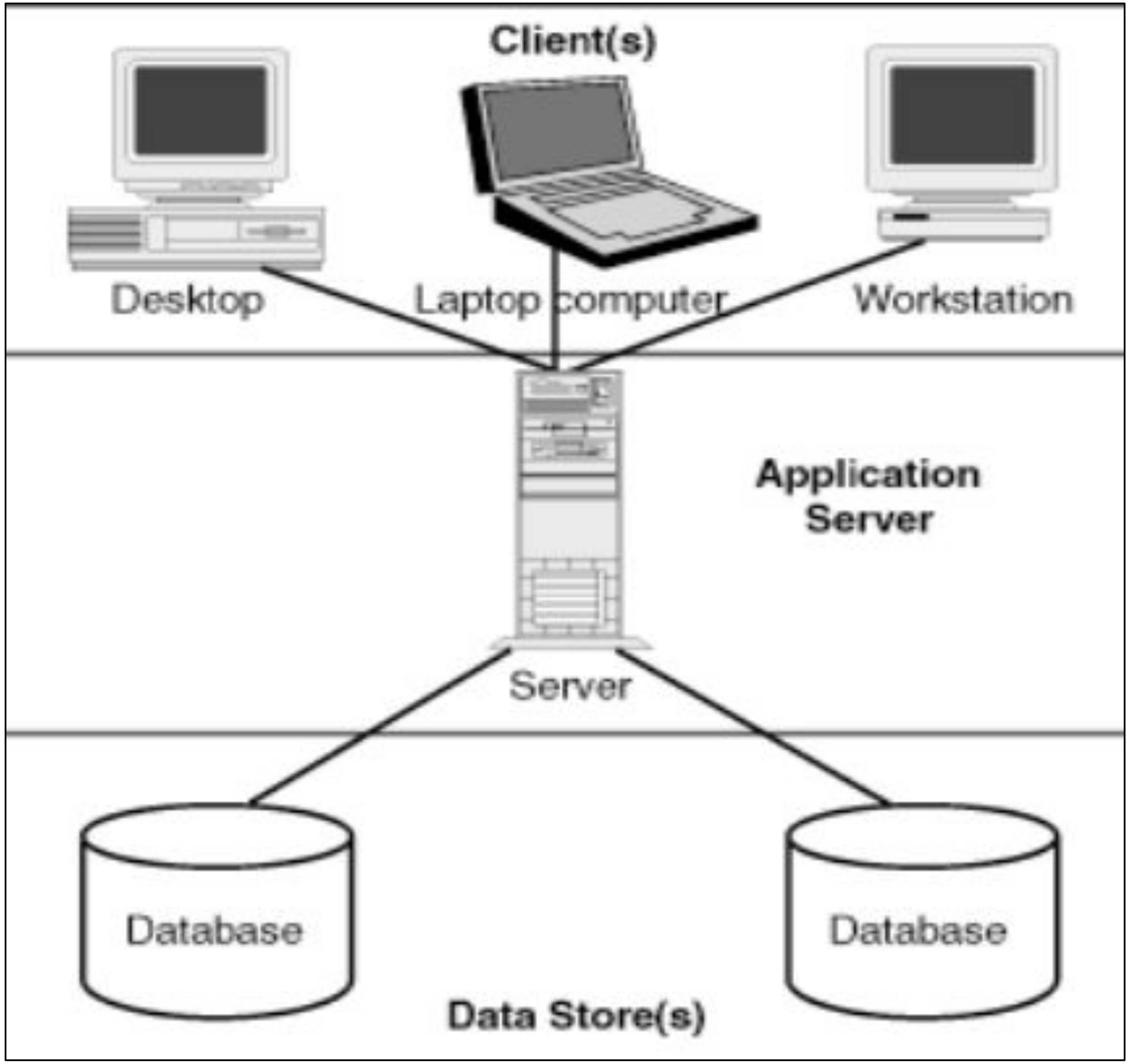
- The program runs on both machines.
- The two-tier architecture is like client server application.
- The direct communication takes place between client and server. There is no intermediate between client and server.
- Relationship between computers as agent / customer.
- This is Message-based infrastructure systems.
- This architecture increases the performance, flexibility of the system



3-tier application model

- There is an additional layer in between client and server that helps clients to get accessed in the database in an efficient way.

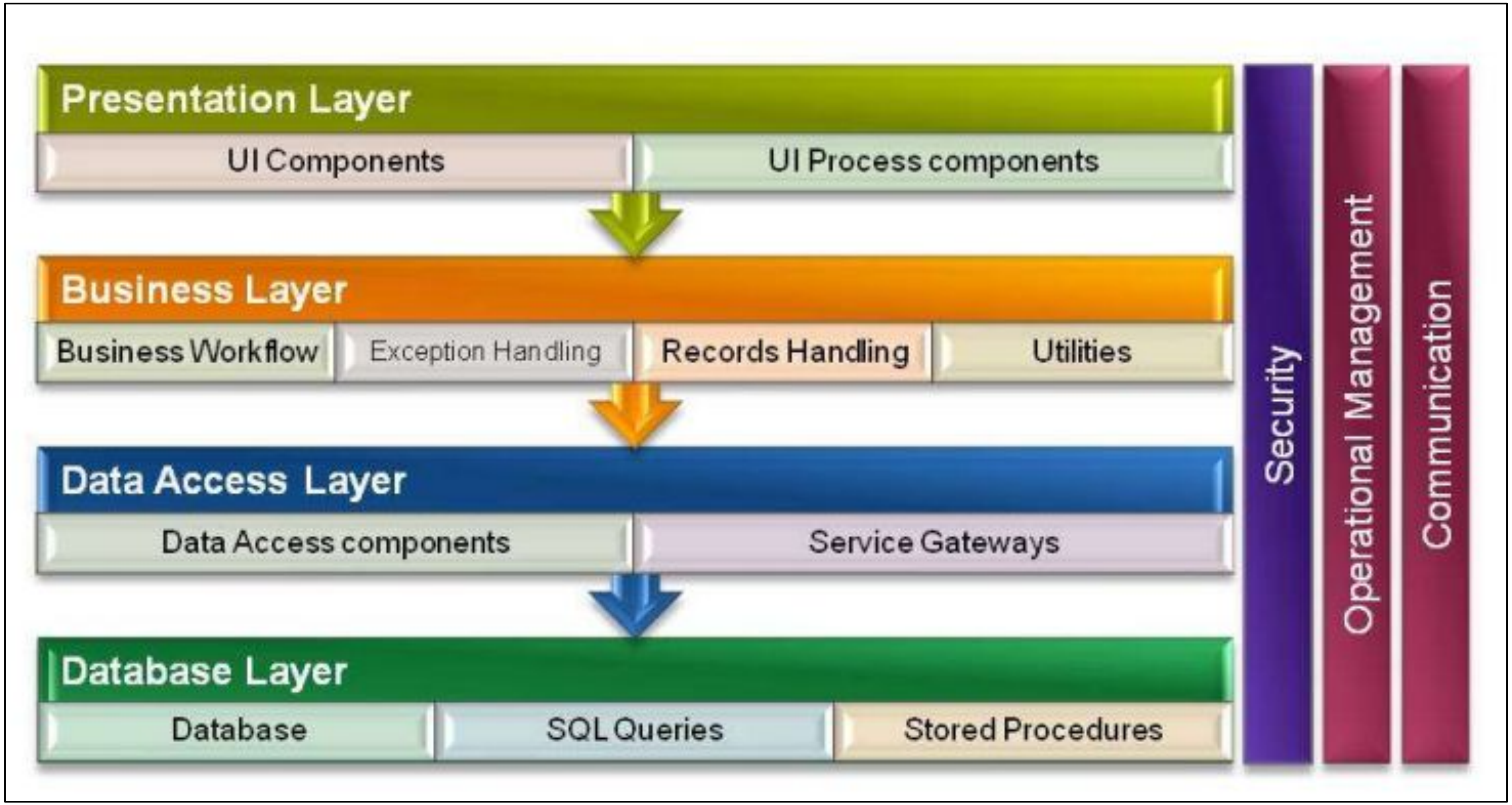
- Breaks down the client server interaction into 3 layers
  1. Application layer / Client tier
  2. Logic layer / Business tier
  3. Storage layer / Data tier


Client(s)
Desktop
Laptop computer
Workstation
Application Server
Server
Database
Database
Data Store(s)

N-tier application model

- Unlimited number of tiers
- Each tier may have multiple computers for operation

- Layers can be
  - Client layer
  - Business logic layer
  - Data access/service layer
  - Data storage layer



Presentation Layer

UI Components

UI Process components

Business Layer

Business Workflow

Exception Handling

Records Handling

Utilities

Data Access Layer

Data Access components

Service Gateways

Database Layer

Database

SQL Queries

Stored Procedures

Security

Operational Management

Communication

Approaches to Designing Client/Server Architectures

- Distributed Presentation
- Remote Presentation
- Remote Data Management
- Distributed Function
- Distributed Database
- Distributed Processing

Designing Internet Systems

- Standards
  - Design is simpler due to standards
  - Naming
    - BIND
  - Translation
    - Hypertext Transfer Protocol (HTTP)
  - Formatting
    - Hypertext Markup Language (HTML)

Designing Internet Systems

- Separating Content and Display
  - HTML has limitations due to format orientation of tags
  - Extensible Markup Language (XML) has been developed to separate content from display
    - Ability to create custom languages
- Future Evolution
  - Move from desktop PCs to thin clients
    - Most processing and data storage occurs on the server

Designing Internet Systems

- Aids to Site Consistency
  - Cascading Style Sheets
    - A set of style rules that tells a Web browser how to present a document
  - Extensible Style Language (XSL)
    - Specification for separating style from content when generating HTML documents

Designing Internet Systems

- Design Issues Related to Site Management
  - Customer Loyalty and Trustworthiness
    - Conveyed by
      - Design quality
      - Up-front disclosure
      - Comprehensive, correct and current content
      - Connected to the rest of the Web
      - Data security
      - Personalization

Designing Internet Systems

- Design Issues Related to Site Management
  - Web Pages Must Live Forever
    - Customer Bookmarks
    - Links from Other Sites
    - Search Engine Referrals
    - Old Content Adds Value
  - System Security

Managing On-line Data

- Effective design in the result of understanding how a system fits within the context of an organization
  - Context Development
- Integration Depth
  - Measurement of how far into the existing technology infrastructure a system penetrates
- Organizational Breadth
  - A measurement that tracks the core business functions affected by a system

Managing On-line Data

- On-line Transaction Processing (OLTP)
  - Refers to immediate automated responses to the requests of users
  - Designed to handle multiple concurrent transactions
  - Plays a large role in electronic commerce applications

Managing On-line Data

- On-line Analytical Processing (OLAP)
    - Refers to graphical software tools that provide complex analysis of data stored in a database.
    - OLAP server is the chief component
    - Good for time series and trend analysis
    - Enables user to “drill-down” into the data

Managing On-line Data

- Comparison of Operational and Informational Systems
    - Operational
        - Interact with customers and run a business in real time
    - Informational
        - Designed to support decision making based on historical data

Managing On-line Data

- Data Warehousing
  - Subject-oriented
    - Organized around key subjects
  - Integrated
    - Data are collected from many operational systems and made to conform to standards
  - Time-variant
    - Data contains a time dimension
  - Nonvolatile
    - Data cannot be updated by users

Managing On-line Data

- Data Warehousing
    - Two primary architectures
        - Two-level
            - Data warehouse and decision support environment
        - Three-level
            - Operational systems
            - Enterprise data warehouse
                - Centralized, integrated data warehouse
                - Control point and single source of all data made available to end users
            - Data marts
                - A data warehouse that is limited in scope based upon aggregation and selection

Extra Contents

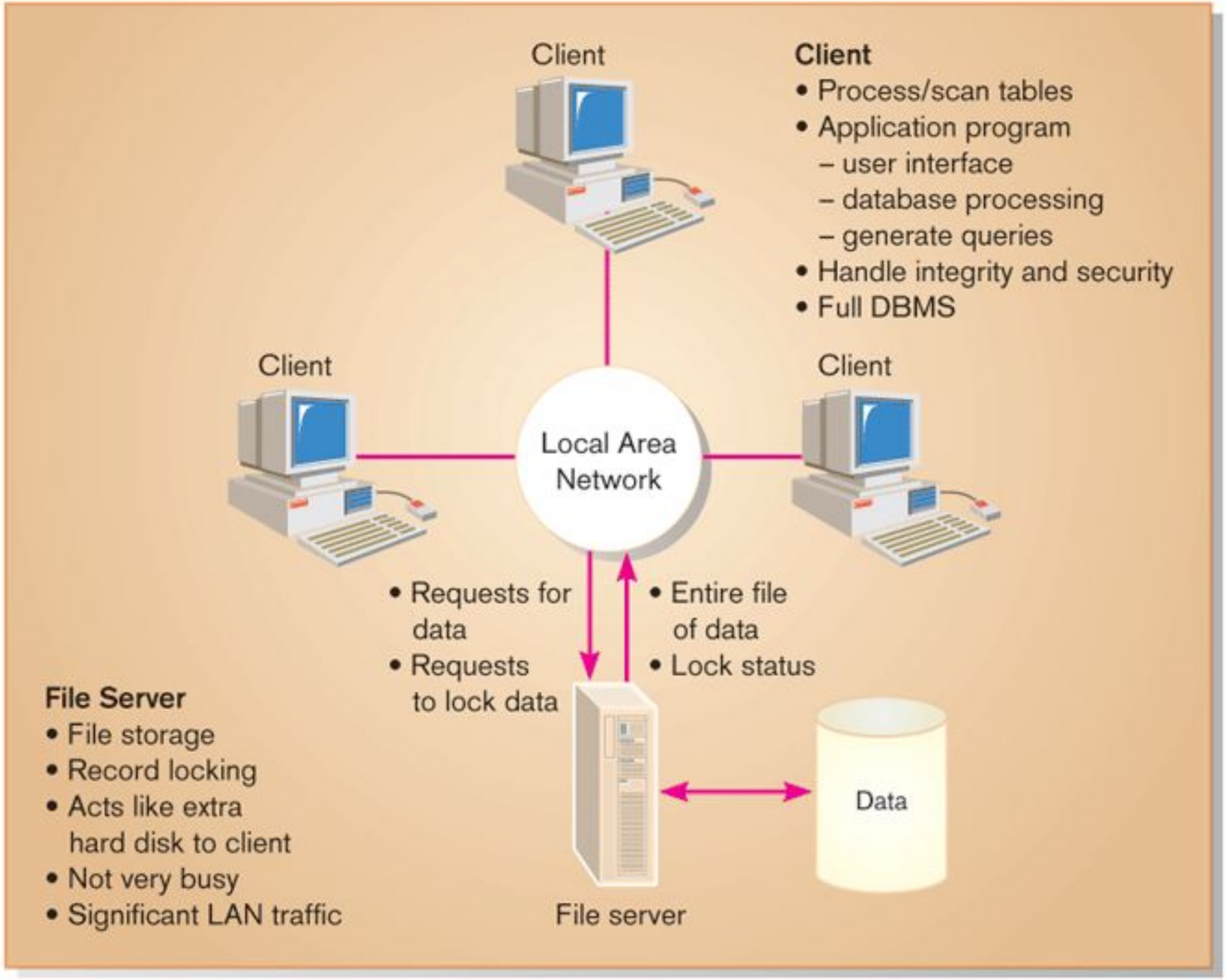
Designing Systems for Local Area Networks (LAN)

- LAN: the cabling, hardware, and software used to connect workstations, computers, and file servers located in a confined geographical area
- Main LAN configuration options
  - File Server architecture
  - Client/Server architecture

File Server Architectures

- A device that manages file operations and is shared by each client PC attached to a LAN
- DBMS use in a file server:
    - One copy of the DBMS is on the file server and concurrently running copies are on client PCs.
    - All data manipulation is performed on the client PC.

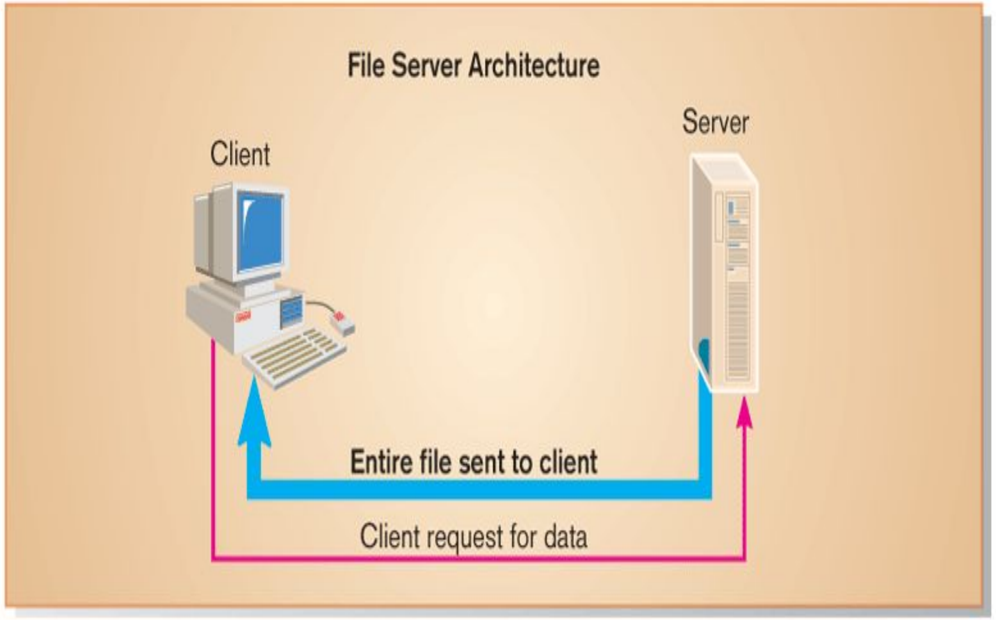
Figure 14-3 File server model



Limitations of File Servers

- Excessive data movement
  - Entire data tables must be transferred instead of individual records
- Need for powerful client workstations
  - Each client workstation must devote memory to a full DBMS
- Decentralized data control
  - Complicates record concurrency control, recovery, and security

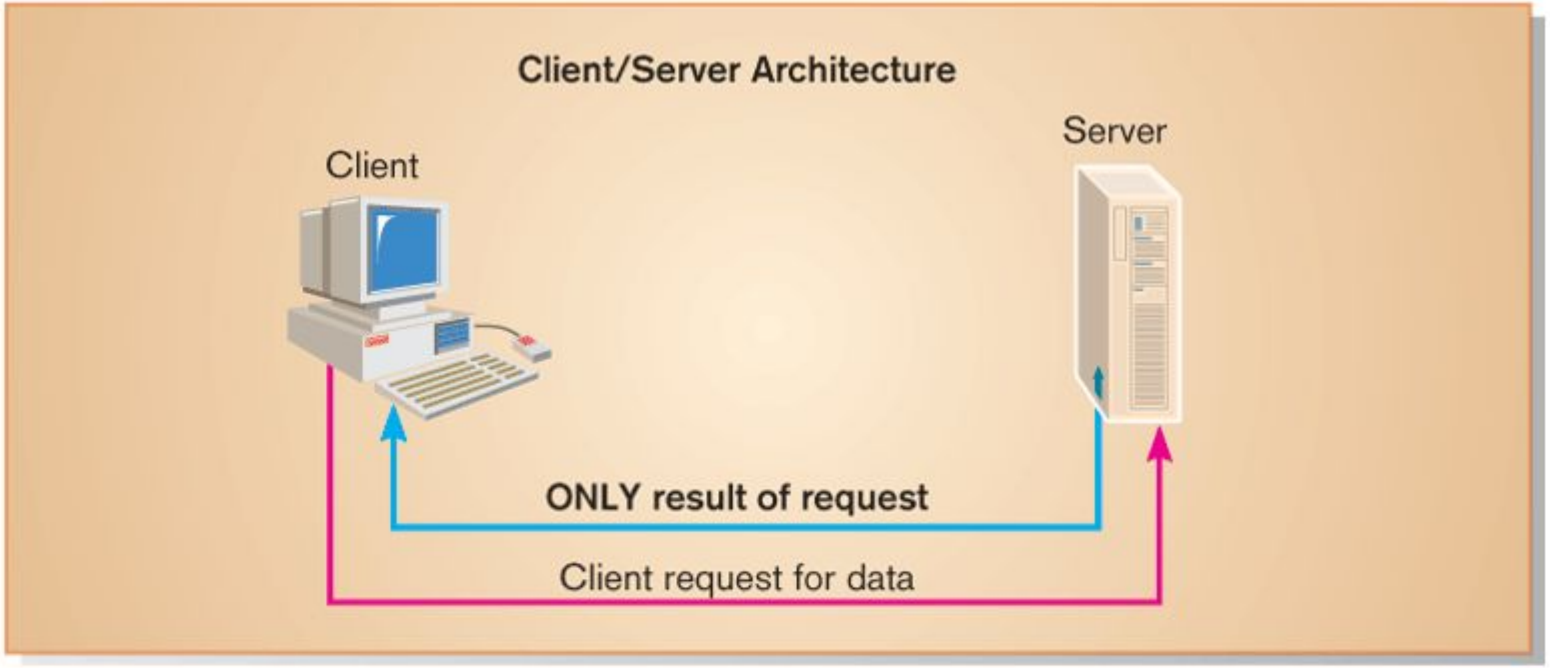
Figure 14-4 File servers transfer files when data are requested from a client



Client/Server Architectures

- Application processing is divided between client and server.
- Client manages the user interface.
- Database server is responsible for data storage and query processing.

Figure 14-5 The required data after a request from a client



Client: front-end software provides user interface and data manipulation functions

Database engine: back-end DBMS software runs on the server to provide database processing and shared access for clients

Application Program Interface (API)

- Software building blocks that ensure standardization of modules for data exchange between clients and servers
- Common API interface can be used for communication between client and any kind of DBMS (DB2, SQL Server, MySQL, Oracle)

File Server vs. Client/Server

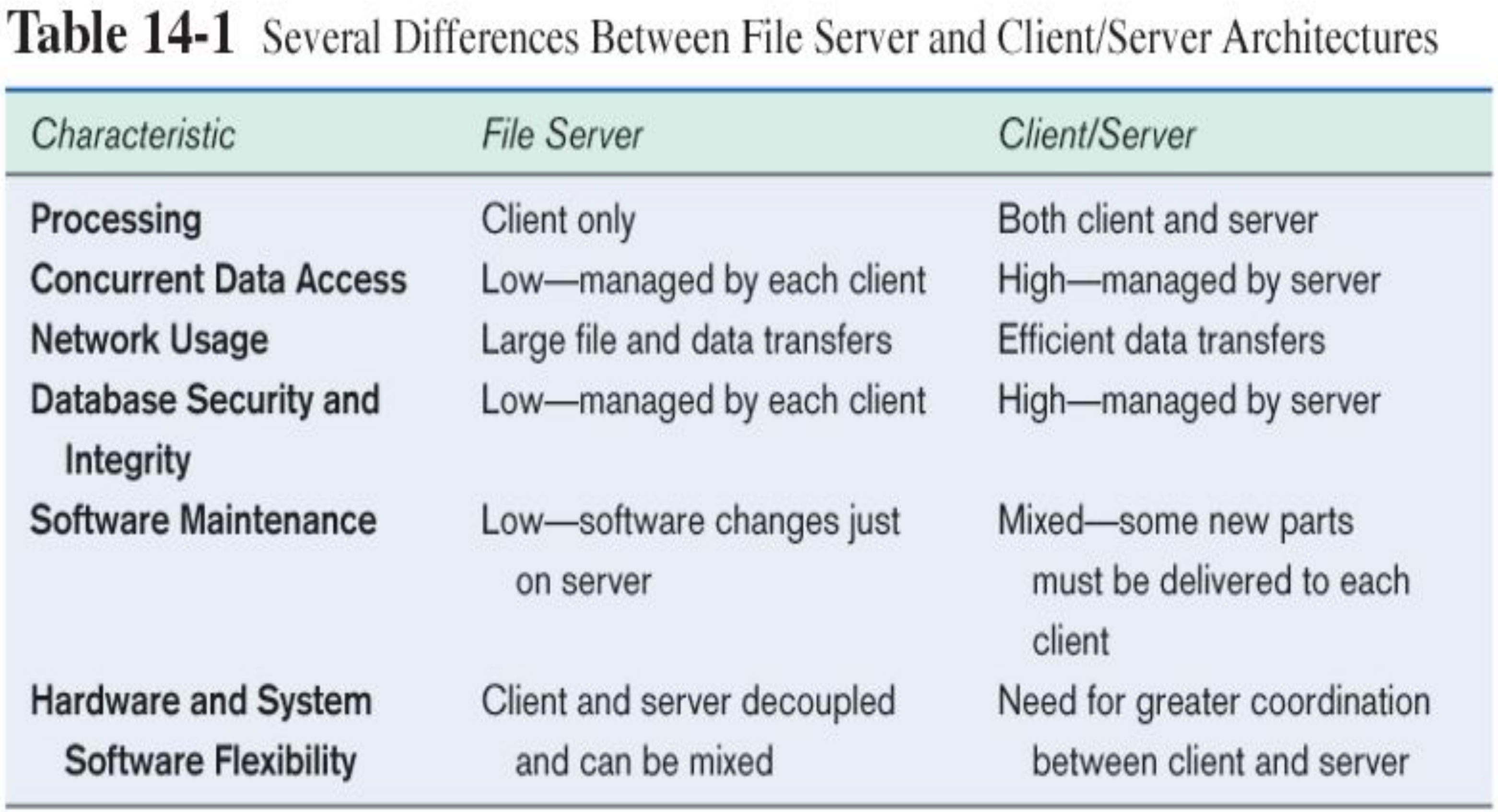


Table 14-1 Several Differences Between File Server and Client/Server Architectures

Characteristic	*File Server*	*Client/Server*
Processing	Client only	Both client and server
Concurrent Data Access	Low—managed by each client	High—managed by server
Network Usage	Large file and data transfers	Efficient data transfers
Database Security and Integrity	Low—managed by each client	High—managed by server
Software Maintenance	Low—software changes just on server	Mixed—some new parts must be delivered to each client
Hardware and System Software Flexibility	Client and server decoupled and can be mixed	Need for greater coordination between client and server

Advanced Forms of Client/Server Architecture

- Three-tiered client/server
  - Three logical and distinct applications
    - Data management
    - Presentation
    - Analysis
  - Middleware
    - Combination of hardware, software, and communication technologies that bring together three distinct applications into one environment
  - Application Server
    - Server where data analysis functions are performed

Advantages of Three-tiered Architectures

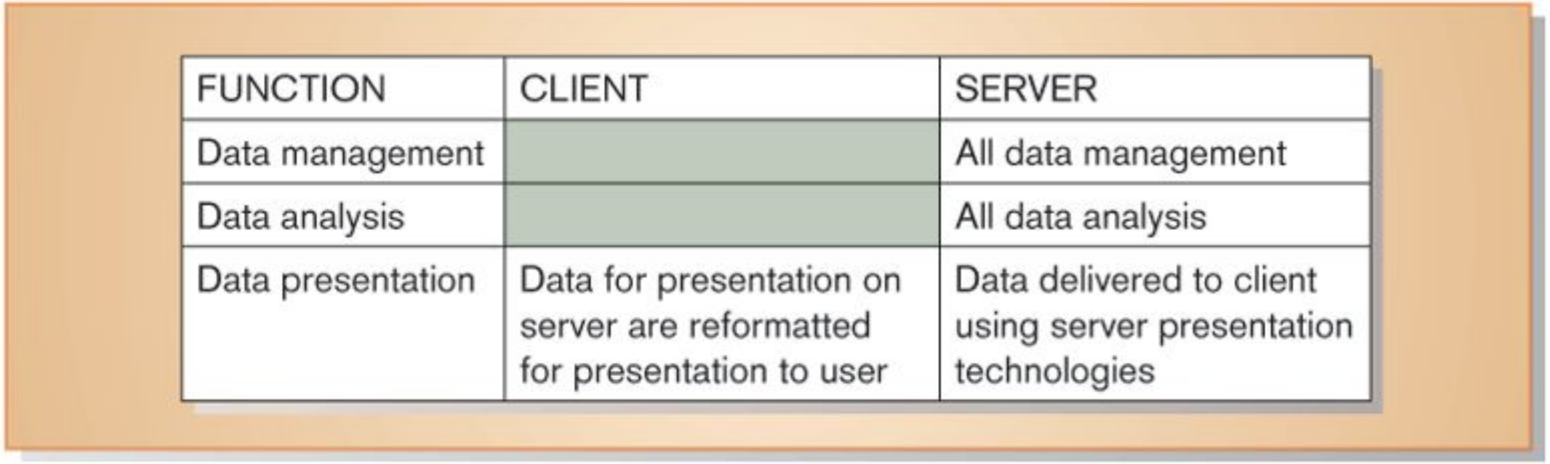
- Applications can be partitioned in a way that best fits the organizational computing need.
- Easier customization: application code resides on application server, so change done only in one place.
- Easier maintenance: data analysis is separate from user interface, so changing one can be done independently of the other.

Approaches to Designing Client/Server Architectures

- Distributed Presentation
- Remote Presentation
- Remote Data Management
- Distributed Function
- Distributed Database
- Distributed Processing

Distributed Presentation

Figure 14-7a Types of client/server architectures - Distributed presentation

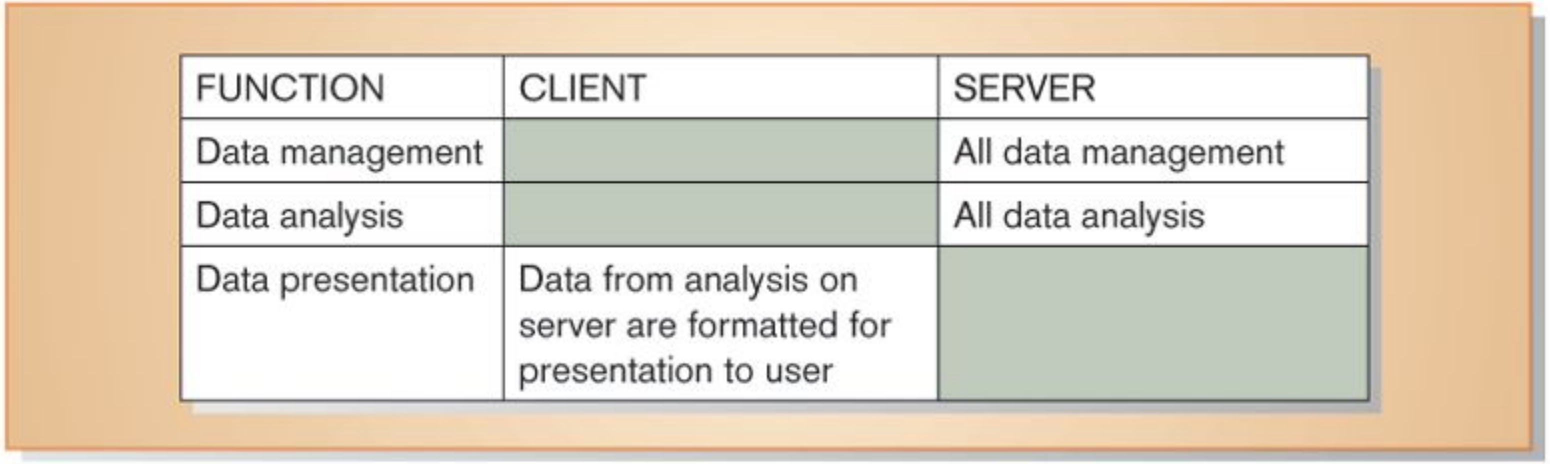


FUNCTION	CLIENT	SERVER
Data management		All data management
Data analysis		All data analysis
Data presentation	Data for presentation on server are reformatted for presentation to user	Data delivered to client using server presentation technologies

Freshen up delivery of existing server-based applications, typically running on legacy mainframe computers, to distributed clients using screen scrapper technology

Remote Presentation

Figure 14-7b Types of client/server architectures - Remote presentation

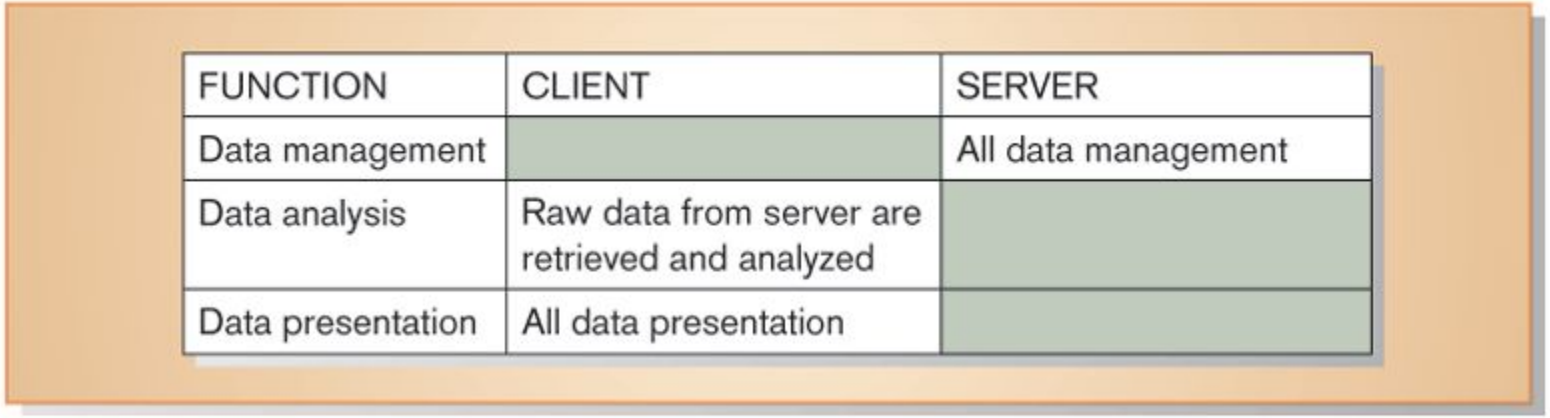


FUNCTION	CLIENT	SERVER
Data management		All data management
Data analysis		All data analysis
Data presentation	Data from analysis on server are formatted for presentation to user	

All data presentation functions are on the client, providing greater flexibility of presentation than the distributed presentation option.

Remote Data Management

Figure 14-7c Types of client/server architectures - Remote data management

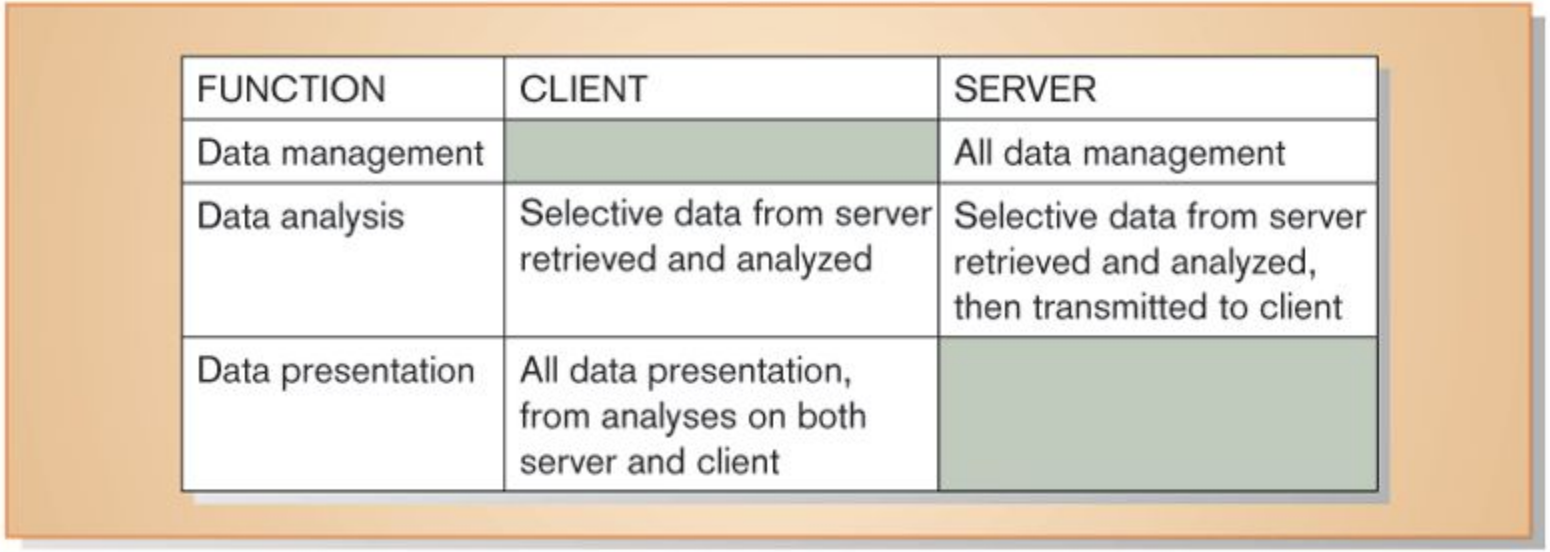


FUNCTION	CLIENT	SERVER
Data management		All data management
Data analysis	Raw data from server are retrieved and analyzed	
Data presentation	All data presentation	

All software except data management is on client, this is closest to the traditional client/server mode.

Distributed Function

Figure 14-7d Types of client/server architectures - Distributed function

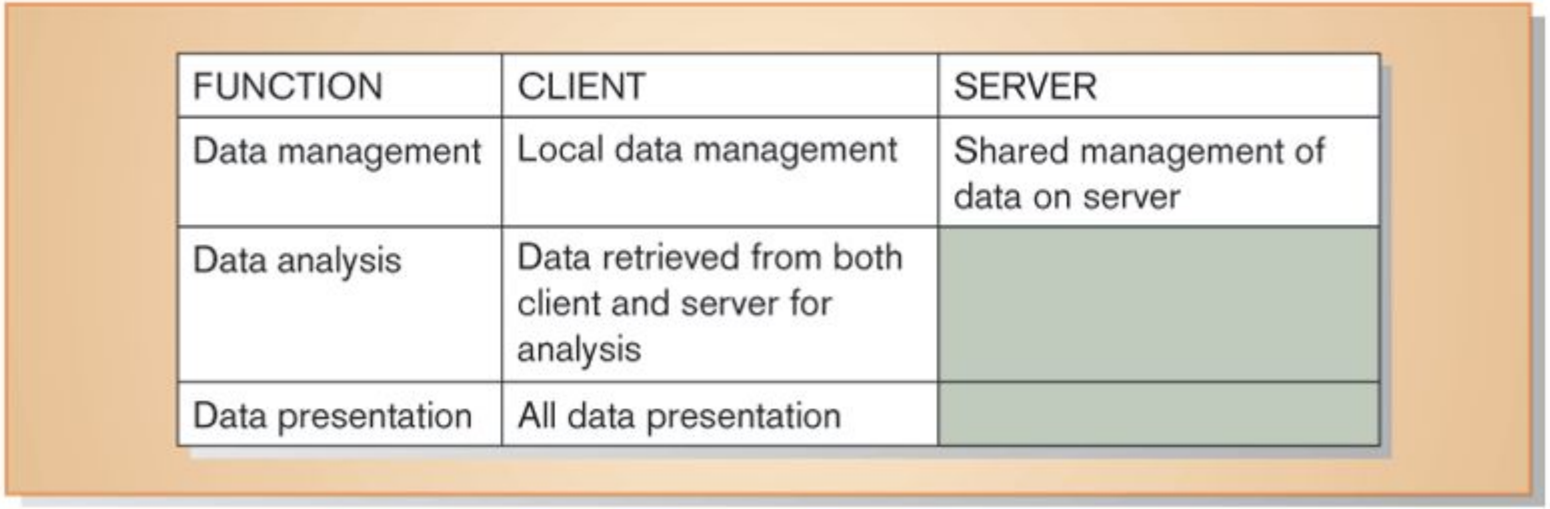


FUNCTION	CLIENT	SERVER
Data management		All data management
Data analysis	Selective data from server retrieved and analyzed	Selective data from server retrieved and analyzed, then transmitted to client
Data presentation	All data presentation, from analyses on both server and client	

Analysis functions are split between client and server, with all presentation on client and all data management on server. Requires coordination between analysis function on client and server, making it difficult to develop and maintain.

Distributed Database

Figure 14-7e Types of client/server architectures - Distributed database

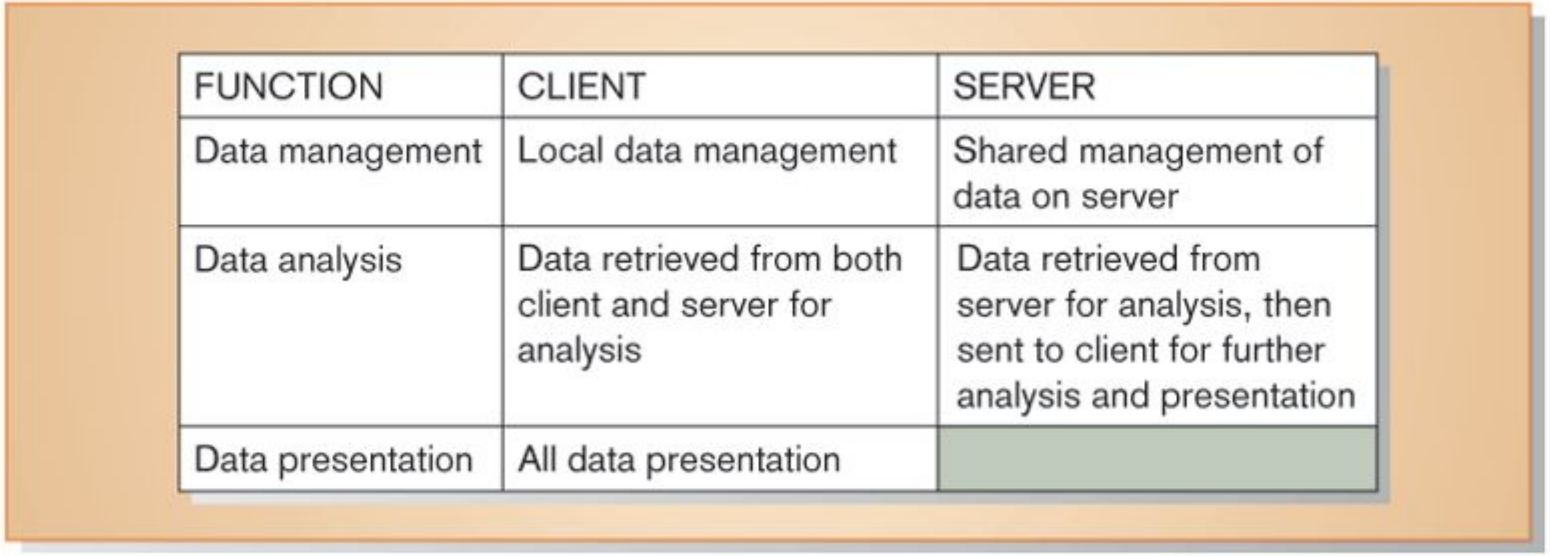


FUNCTION	CLIENT	SERVER
Data management	Local data management	Shared management of data on server
Data analysis	Data retrieved from both client and server for analysis	
Data presentation	All data presentation	

Client has all functionality, except that data storage and management is shared between client and server. A distributed database is unstable, and it is very difficult to ensure compatibility and communication between client and server.

Distributed Processing

Figure 14-7f Types of client/server architectures - Distributed processing



FUNCTION	CLIENT	SERVER
Data management	Local data management	Shared management of data on server
Data analysis	Data retrieved from both client and server for analysis	Data retrieved from server for analysis, then sent to client for further analysis and presentation
Data presentation	All data presentation	

Combines distributed function and distributed database, maximizing flexibility of analysis and data management