Lecture 6.a
ICT1 Review

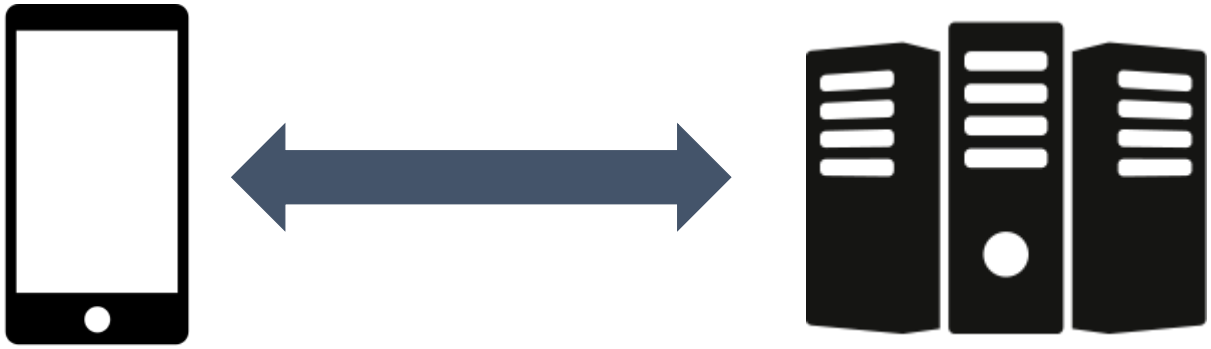
Client-Server Architecture

Dr. Gabriele Pierantoni

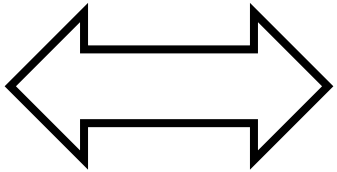
22.02.2021

Local and Remote Programming

- Up to know, you have only studied programs that run in your local machine but most of the programs you use in your daily life does not run on your local phone/laptop.



Client-Server Paradigm

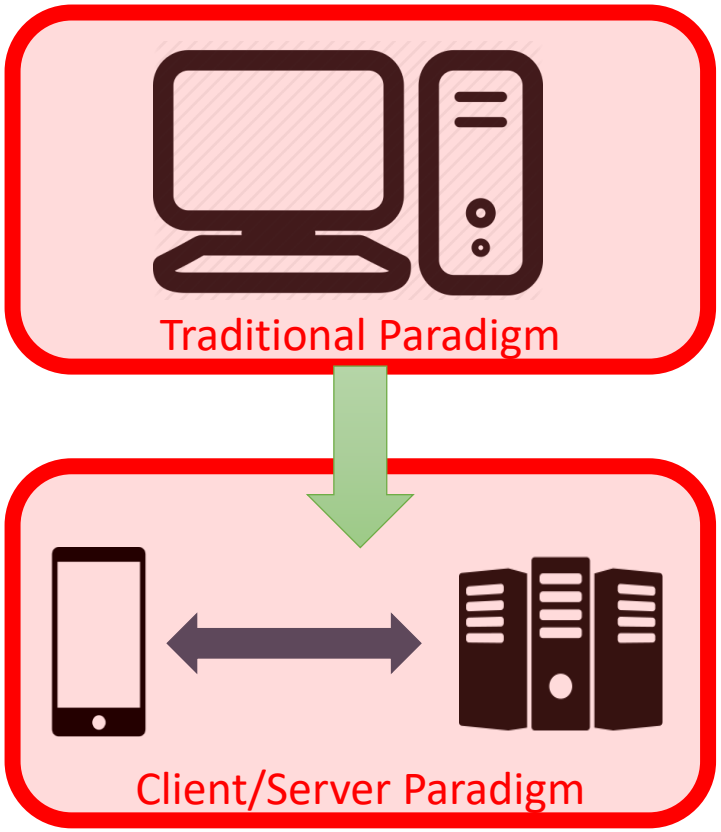


Client

Server(s)

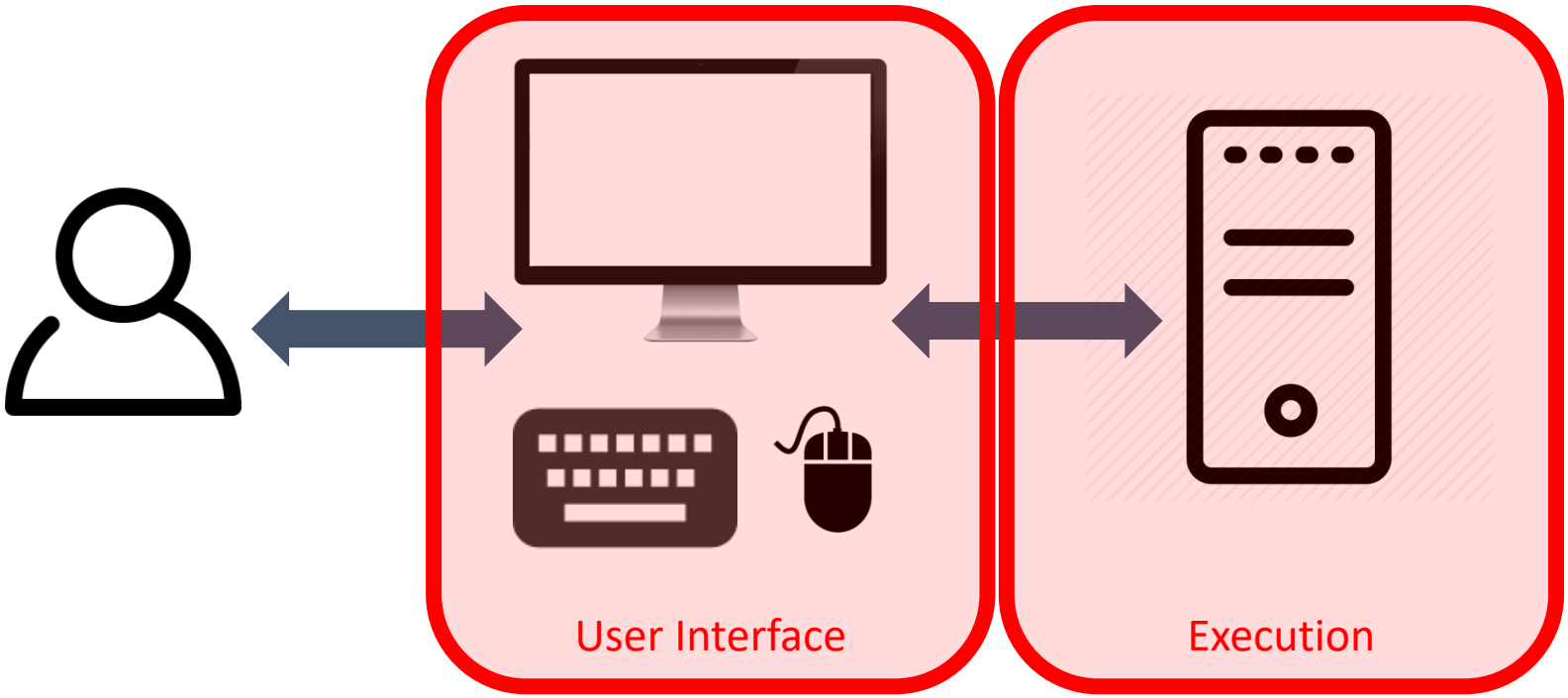
Client-Server Paradigm

Whenever you move a program (or part of a program) from the device it is accessed from (traditional architecture) to a client-server architecture, an entire new domain of programming opens up that covers many facets.



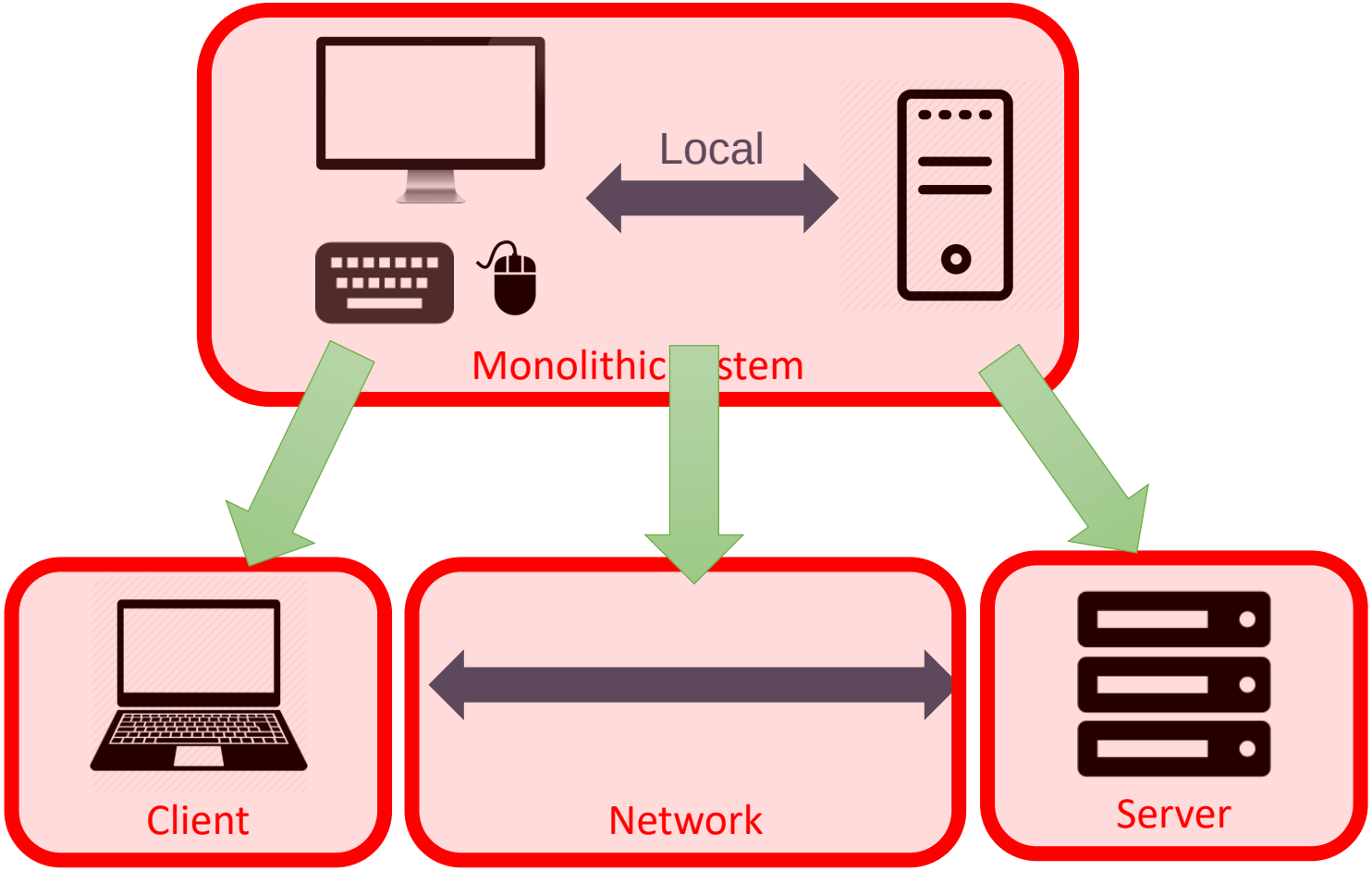
Traditional Programming

The interactions between the user and the system and the execution of the program all happen in the same system. That is easy !



Client-Server Programming (2)

Let's separate the user interface and the main execution of the program, i.e. two basic components of the system.

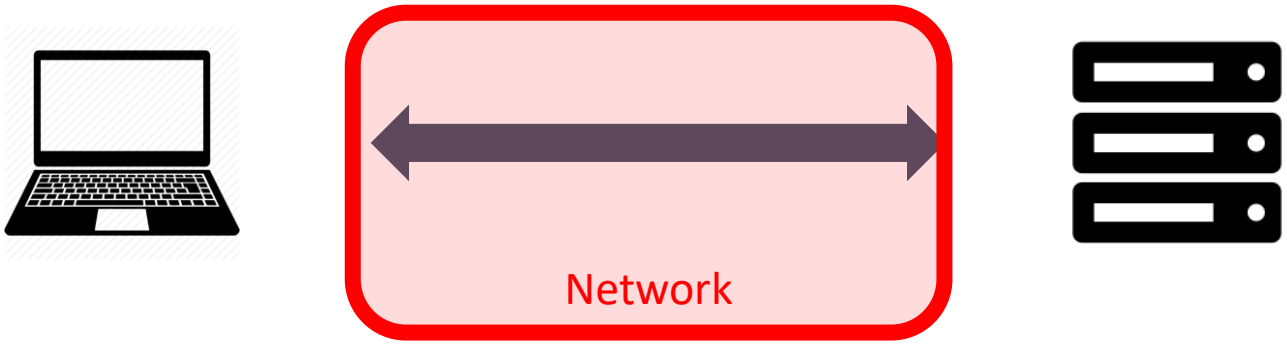


Client-Server Programming (3)

Clients: how to program the client-side.

Client-Server Programming (4)

Network: how to program the network-side



Client-Server Programming (5)

Server: how to program the server-side.

Computing Models

Computing Models

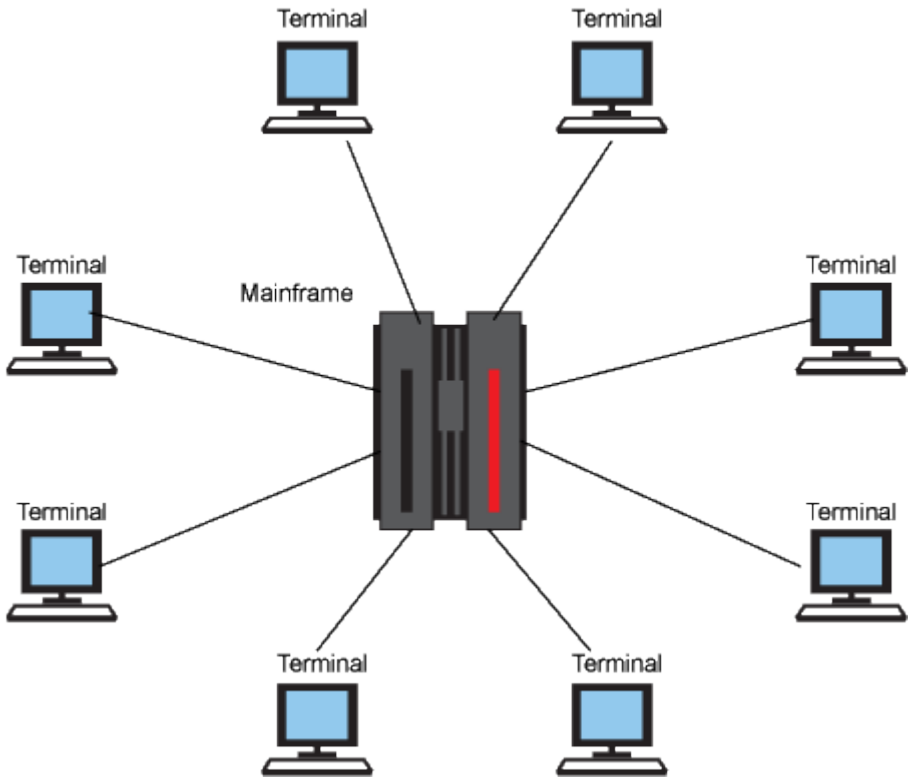
System architecture

- High-level design on which the system is based
- It contains
  - Components
  - Collaborations (how components interact)
  - Connectors (how components communicate)

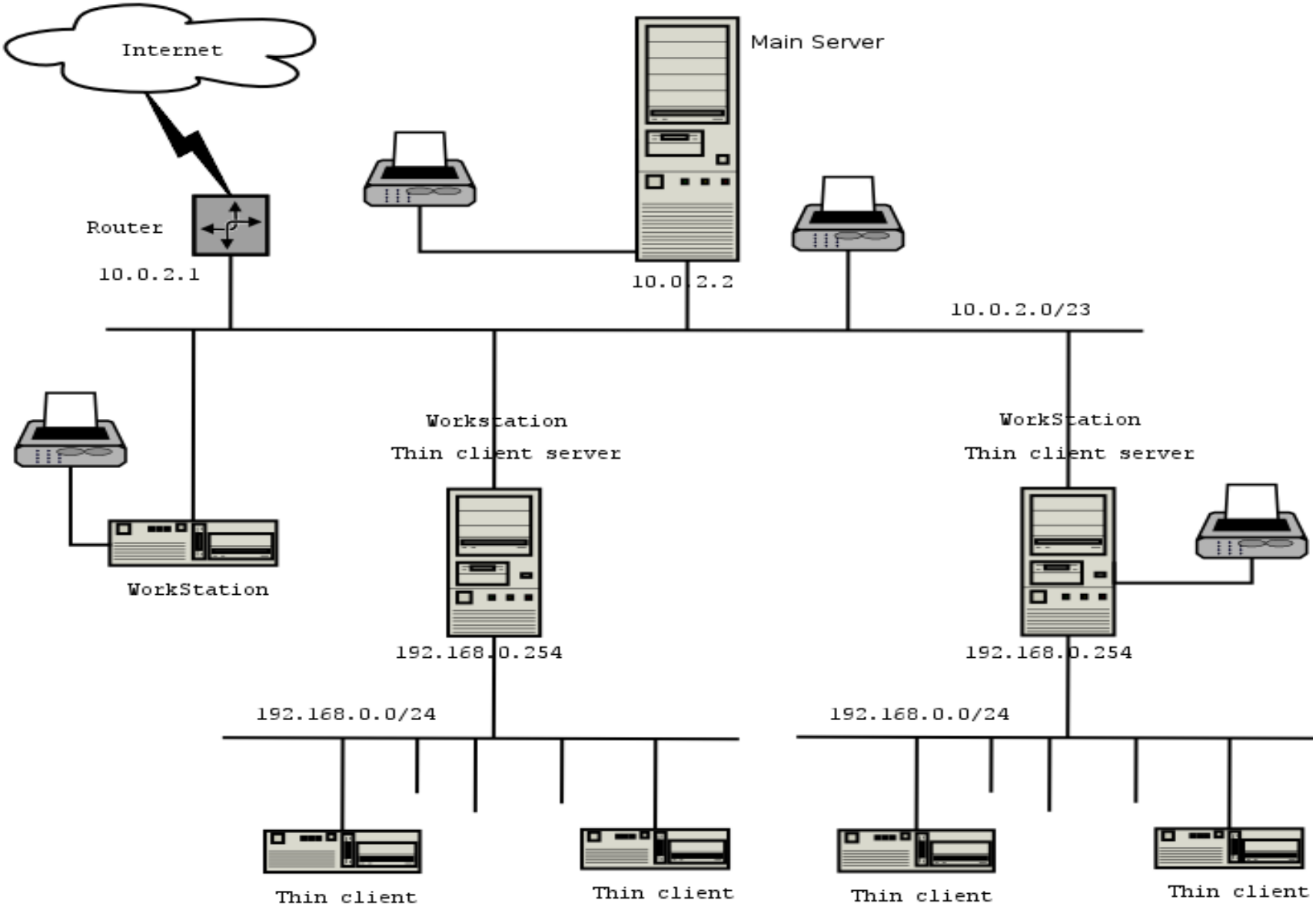
Computing models

- Centralized (or mainframe) model
- Distributed model

Mainframe Model



Distributed Model



Client-Server Paradigm

Client-Server Paradigm (1)

Definition (wiki)

- **It is based on the distributed model that partitions tasks or workloads between the providers of a resource or service (server) and service requesters (client) that communicate over computer network.**
- **A Server host runs one or more server programs which share their resources with clients**
- **A Client does not share any of its resources, but requests a server's content or service function**
- **The Server accepts Requests from the Client and returns the result to the Client**
- **The Client-Server Architecture separates a program from the device it is accessed from**

Client-Server Paradigm (2)

Components

Clients: **are active initiators of transactions making requests for services,**

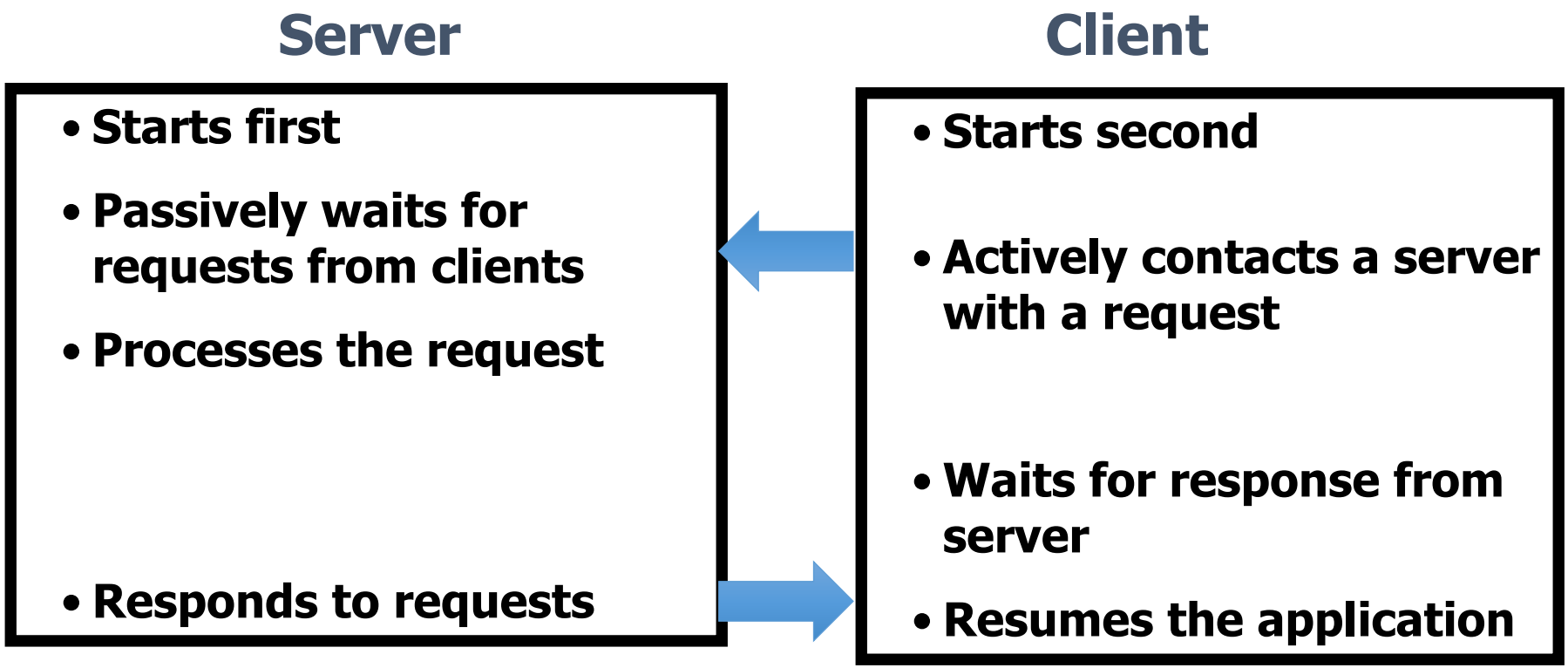
- **they are user point of entry into a network,**
- **they normally execute in a desktop computer, workstation, or laptop (or phone),**
- **a User generally interacts directly only with the Client portion of an application.**
- **example: Web Browser**

Servers: **they manage Hardware and Software resources to process requests**

- **Servers are passive and react to Client requests**
- **example: web server**

communication network: **connects clients and servers through computer networks**

Client-Server Paradigm (3)



Client-Server Architecture

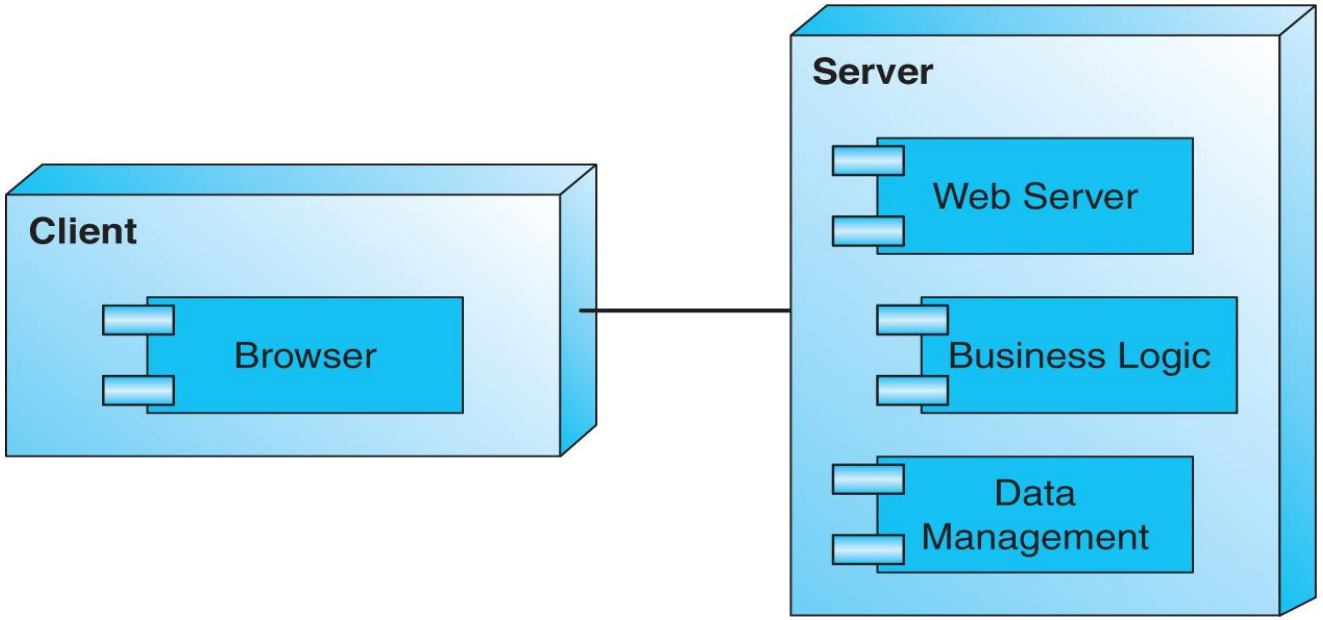
Client-Server Architecture: 2-Tier Architecture (1)

–Tier 1:

- **It comprises the Client Platform (example: hosting a Web Browser).**

–Tier 2:

- **It comprises the Server Platform, hosting all Server Software components**



Tier 1: **Tier 2:**

Client-Server Architecture: 2-Tier Architecture (2)

- **Typical application**
  - **10-100 users**
  - **Small company or organization, e.g., law office, medical practice, local non-profit**
- **Advantage:**
  - **Inexpensive (single platform)**
- **Disadvantages**
  - **Interdependency (coupling) of components**
  - **No redundancy**
  - **Limited scalability**

Client-Server Architecture: 3-Tier Architecture (1)

–Tier 1:

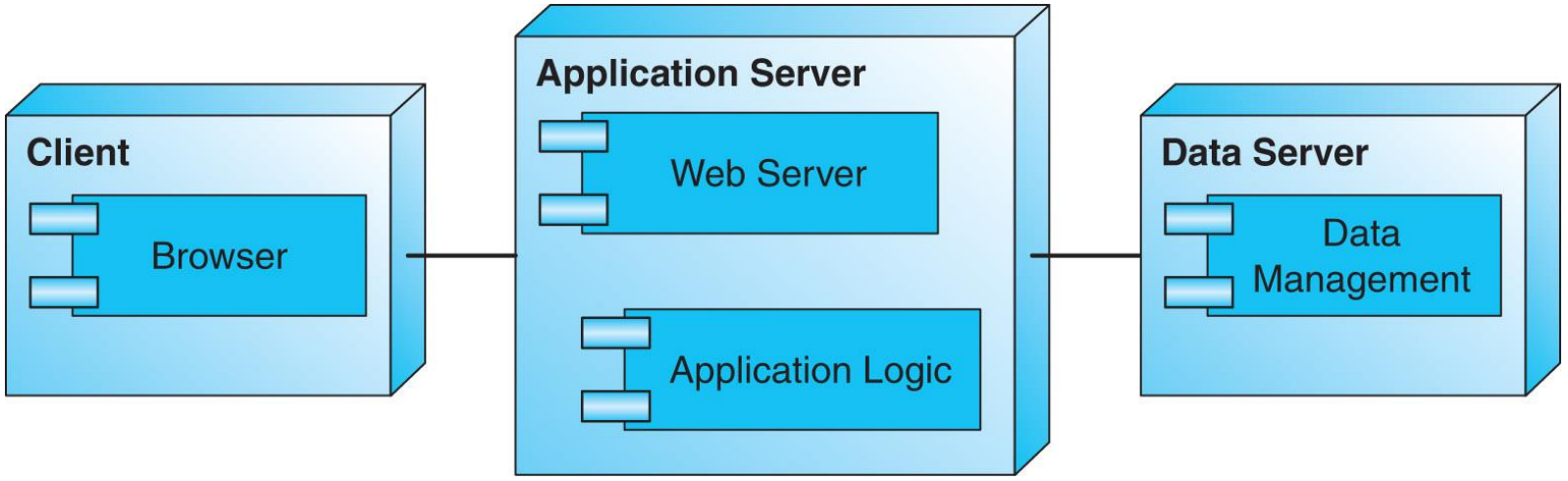
- **It comprises the Client Platform (example: hosting a Web Browser).**

–Tier 2:

- **It comprises the Application Server Platform hosting application logic and web server**

–Tier 3:

- **It comprises the Data Server platform**



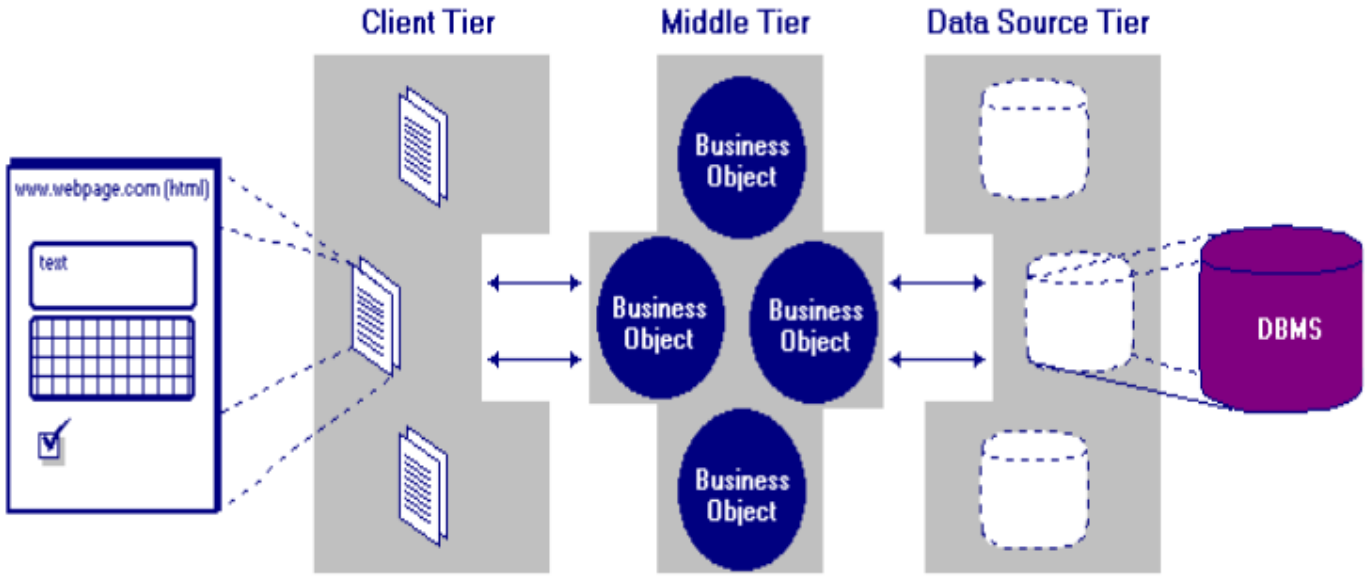
Tier 1: **Tier 2:** **Tier 3:**

Client-Server Architecture: 3-Tier Architecture (2)

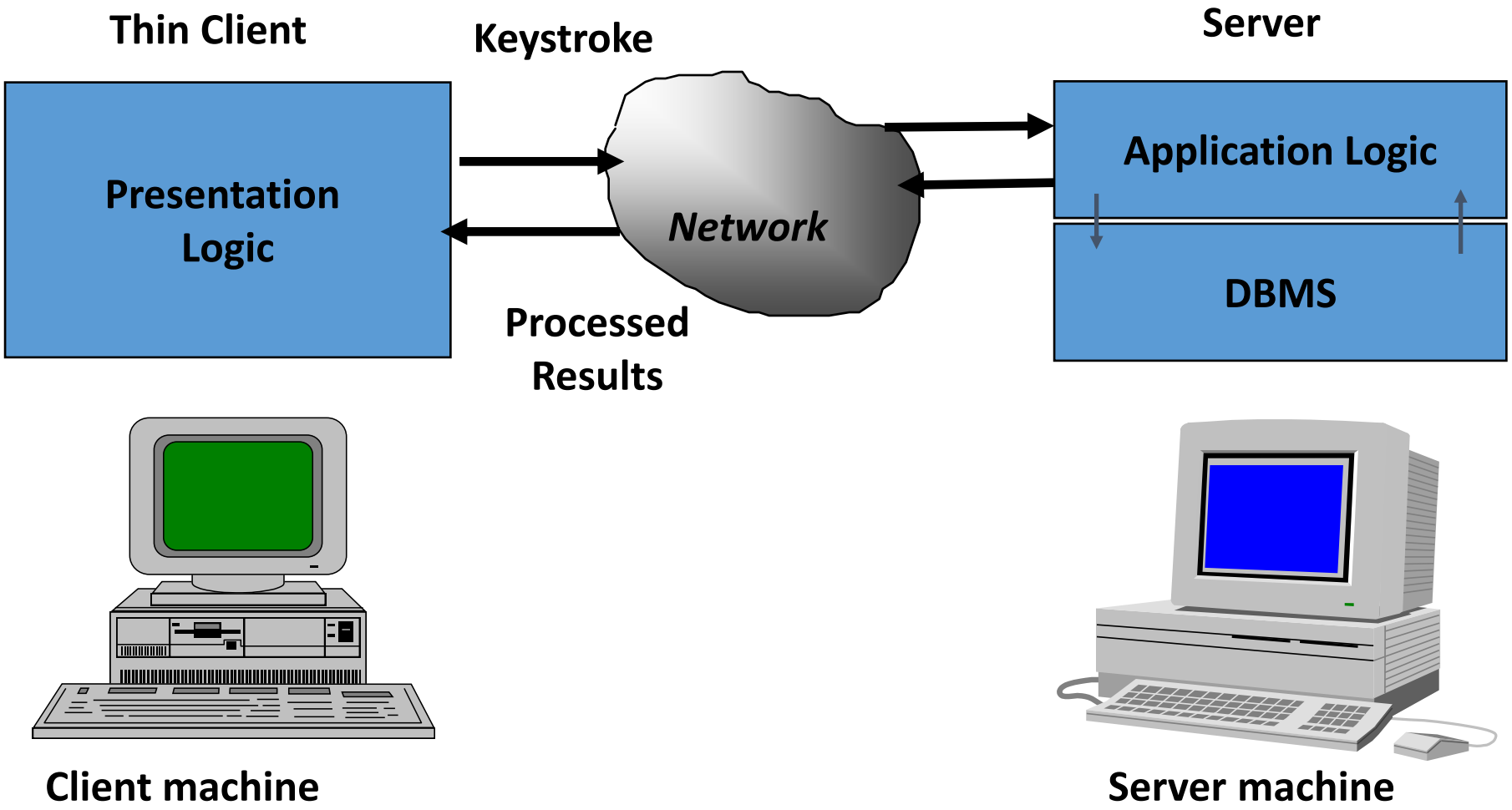
- **Typical Application**
  - **100-1000 users**
  - **Small business or regional organization, e.g., specialty retailer, small college**
- **Advantages**
  - **Improved performance, from specialized hardware**
  - **Decreased coupling of software components**
  - **Improved scalability**
- **Disadvantages**
  - **No redundancy**

Client-Server Architecture: Multi-Tier Architecture

- **Expansion of the 3-tier architecture, in one of several different possible ways:**
  - **Replication of the function of a tier**
  - **Specialization of function within a tier**
  - **Portal services, focusing on handling incoming web traffic**

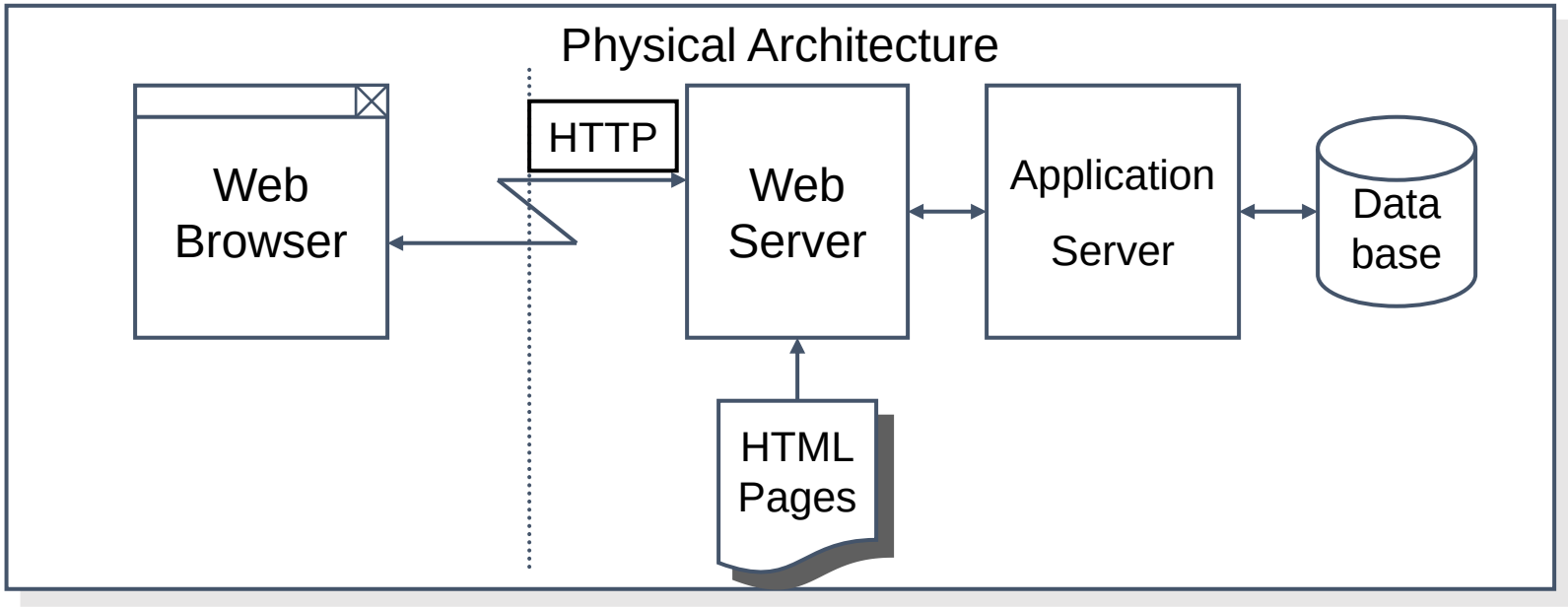


Client-Server Architecture: Thin Client Model (1)

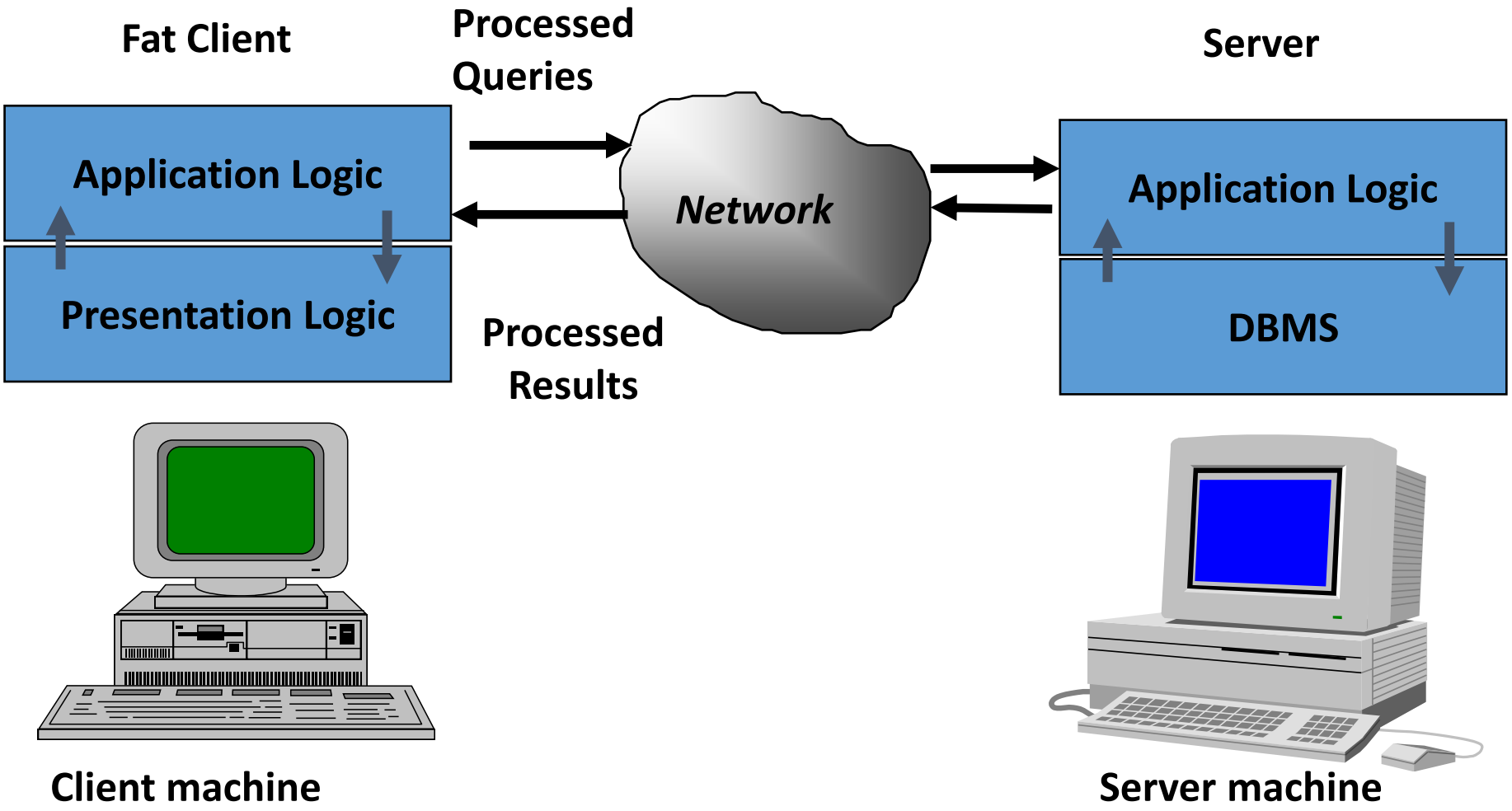


Client-Server Architecture: Thin Client Model (2)

- **No extra software distribution required on the client computer**
- **It connects to server for every little action (e.g. input validation)**
- **Offers a limited user interface design options with semi-static GUI**

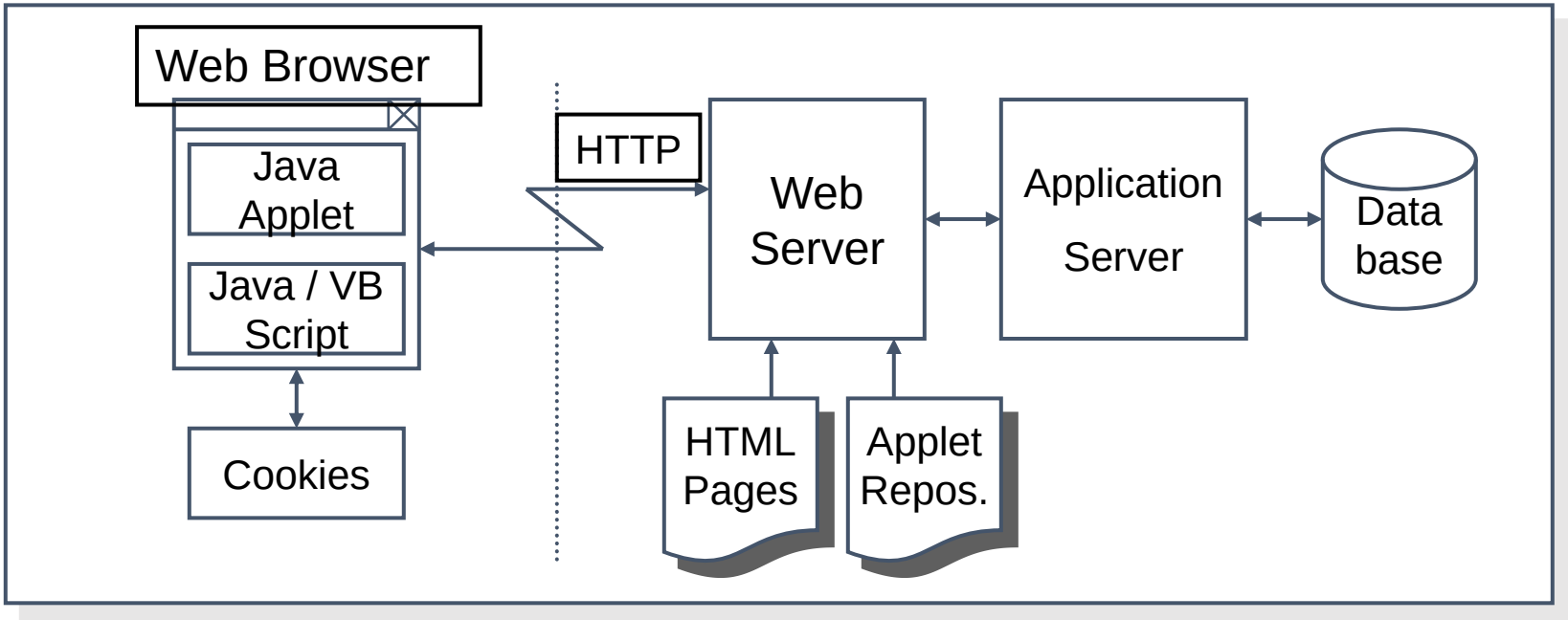


Client-Server Architecture: Fat Client Model (1)



Client-Server Architecture: Fat Client Model (2)

- **Automatic software distribution with software deployed on the client computer**
- **Offers some functionalities without connection**
- **It allows for active and richer GUIs with immediate response**



Client-Server Middleware

Client-Server Middleware (1)

Middleware:

- **Software which allows an application to interoperate with other software, without requiring the user to understand and code the low-level operations required to achieve interoperability**

- **Types of interoperation:**
  - **Synchronous Systems - the requesting system waits for a response to the request in real time**

  - **Asynchronous Systems - send a request but do not wait for a response in real time – the response is accepted whenever it is received .**

Client-Server Middleware (2)

Asynchronous Remote Procedure Calls (RPC)

- **Client makes calls to procedures running on remote computers but does not wait for a response. If connection is lost, client must re-establish the connection and send request again.**

Synchronous RPC

- **Distributed program using this may call services available on different computers – makes it possible to achieve this without undertaking detailed coding (e.g. RMI in Java)**

Publish/Subscribe

- **Push technology - server monitors client activities and sends information to the client when available.**
- **Asynchronous, the clients (subscribers) perform other activities between notifications from the server.**
- **Useful for monitoring situations where actions need to be taken when particular events occur.**

Client-Server Middleware (2)

Asynchronous Remote Procedure Calls (RPC)

- **Client makes calls to procedures running on remote computers but does not wait for a response. If connection is lost, client must re-establish the connection and send request again.**

Synchronous RPC

- **Distributed program using this may call services available on different computers – makes it possible to achieve this without undertaking detailed coding (e.g. RMI in Java)**

Publish/Subscribe

- **Push technology - server monitors client activities and sends information to the client when available.**
- **Asynchronous, the clients (subscribers) perform other activities between notifications from the server.**
- **Useful for monitoring situations where actions need to be taken when particular events occur.**

Client-Server Middleware (3)

Message-Oriented Middleware (MOM)

- **Asynchronous – sends messages that are collected and stored until they are acted upon, while the client continues with other processing.**

Object Request Broker (ORB)

- **Object-oriented management of communications between clients and servers.**
- **ORB tracks the location of each object and routes requests to each object.**

SQL-oriented Data Access

- **Middleware between applications and database servers. Has the capability to translate generic SQL into the SQL specific to the database**

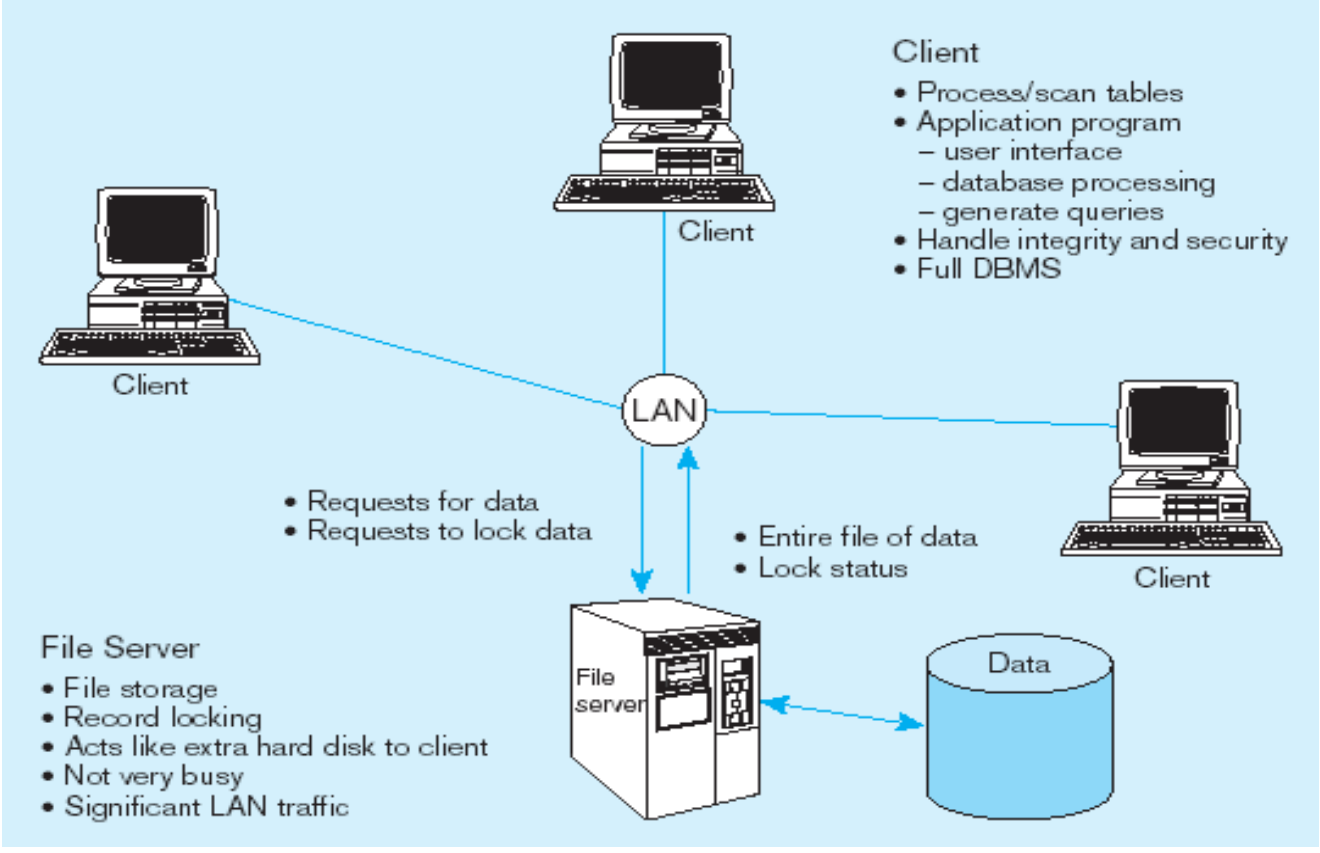
Client-Server Architecture: Server Types

Client-Server Architecture: Server Types

- File and print
- Web
- Proxy
- Caching
- Mail
- Mailing list
- Media
- DNS
- FTP
- News
- Certificate
- Directory
- Catalog
- Transaction

File Servers

- **They manage client application files providing shared access**
- **They pull large amount of data off the storage subsystem and pass the data over the network**
- **They require many slots for network connections, large-capacity and fast hard disk subsystem**



Computer Servers

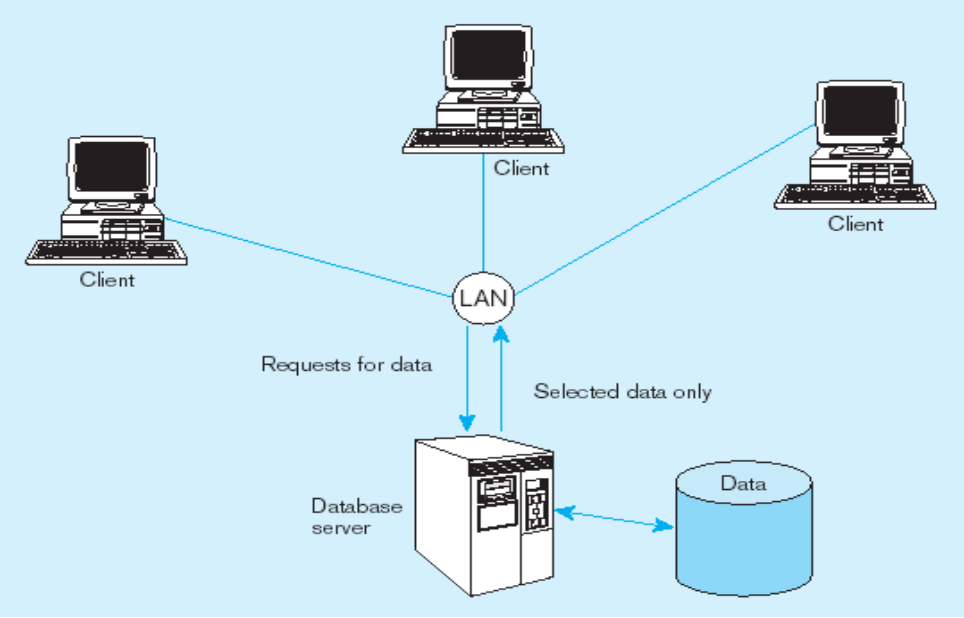
- **Performs application logic processing**
- **Compute servers requires**
  - **processors with high performance capabilities**
  - **large amounts of memory**
  - **relatively low disk subsystems**
- **By separating data from the computation processing, the compute server's processing capabilities can be optimized**

Data Servers

- **Data-oriented used only for data storage and management and does not perform any application logic**
- **Perform processes such as data validation, required as part of the data management function.**
- **Since a data server can serve more than one compute server, compute-intensive applications can be spread among multiple severs**
- **Requires fast processor, large amount of memory and substantial hard disk capacity.**

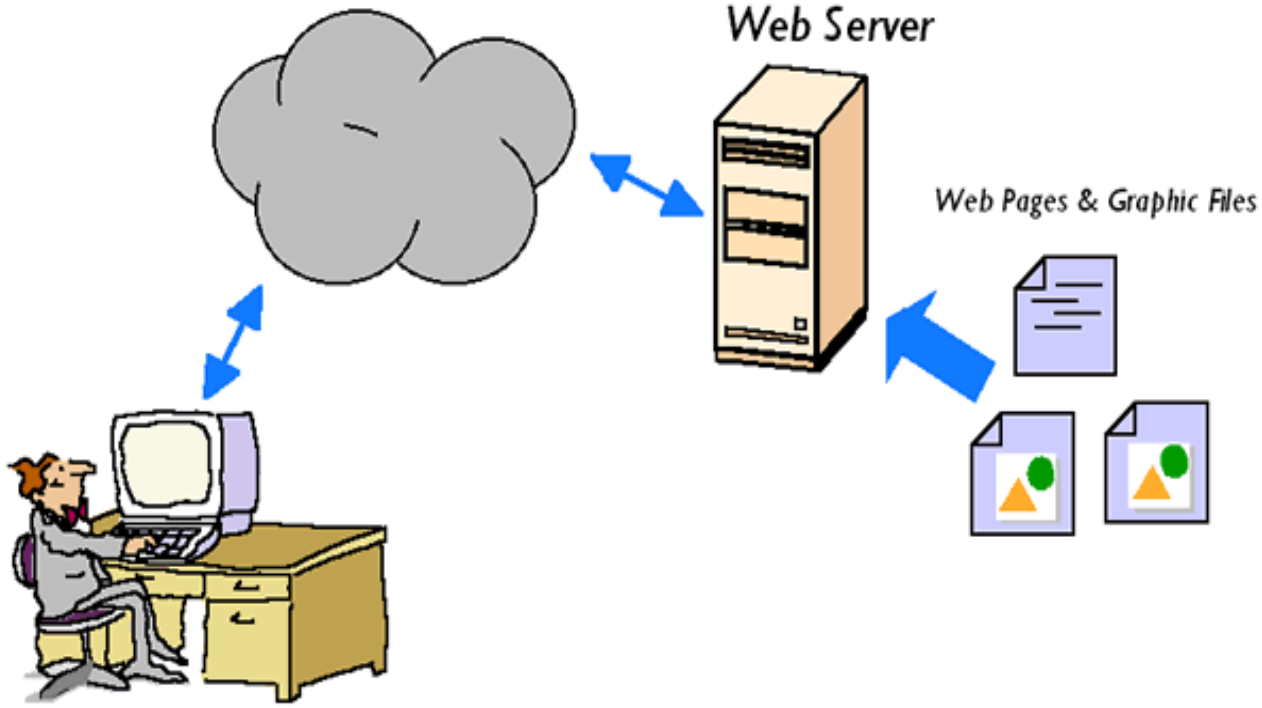
Database Servers

- **They accept requests for data, retrieve the data from its database (or requests data from another node) and pass the results back.**
- **The server requirement depends on the size of database, speed with which the database must be updated, number of users and type of network used.**



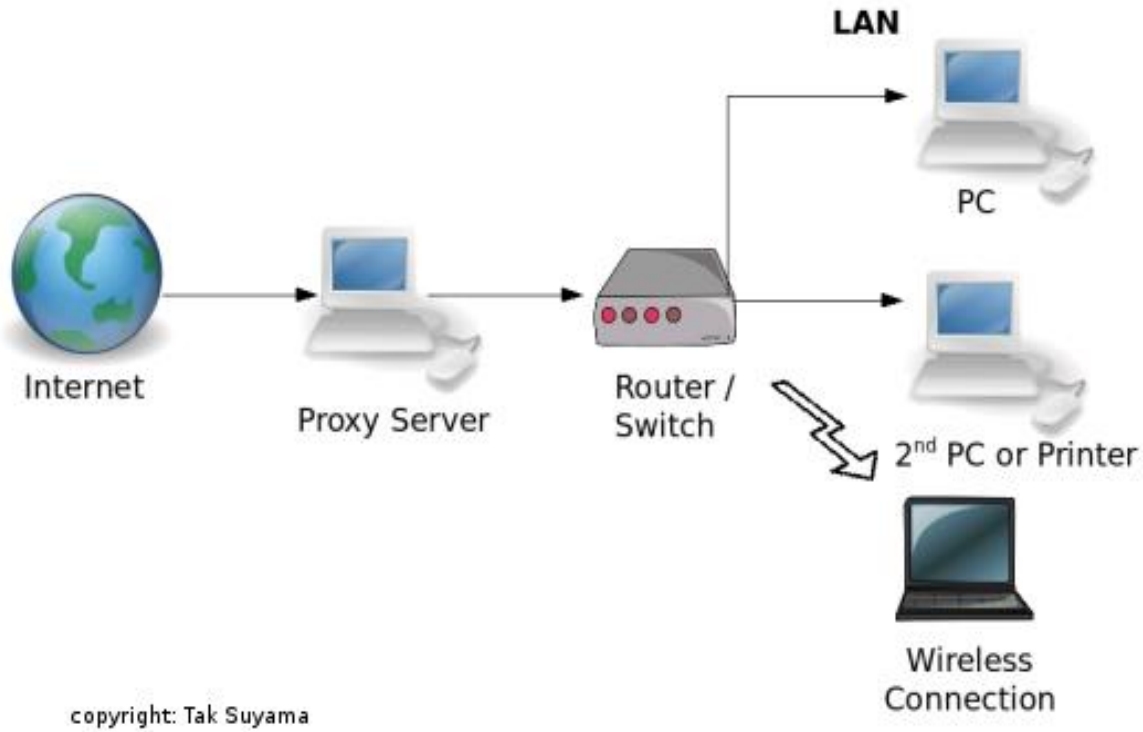
Web Servers

- They store, process and deliver web pages to clients using the HTTP protocol
- Web Browser initiates communication sending a request for a resource using HTTP and the server responds with the content
- They run presentation functions such as:
  - gathering and validating inputs
  - request routing
  - authentication
  - load balancing
  - database routing



Proxy Servers

- They are machines which act as an intermediary between the computers of a local area network and the Internet
- They deliver the following features:
  - caching Web documents
  - providing corporate firewall access
  - filtering client transactions
  - logging transactions
  - securing the host
  - enabling enhanced admin



Mail Servers

- **They are high-capacity computing devices that run software dedicated to sending, delivery, and storage of electronic mail messages.**
- **Their software allows the sysadmin to create and manage email accounts**
- **They send and receive email using standard email protocols for example, the SMTP protocol sends messages and handles outgoing mail requests. The IMAP and POP3 protocols receive messages and are used to process incoming mail**