Internet and Intranet

Lecture by:

Jalauddin Mansur

June 2015

Chapter 3:Protocols and Client/Server Applications

Topics:

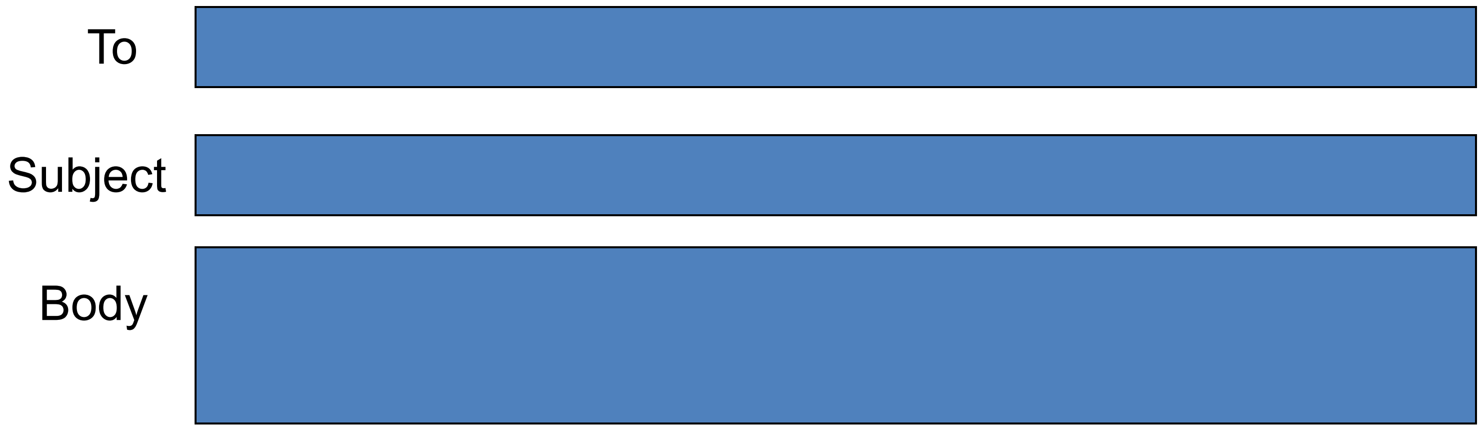
- Standard Protocols: SMTP, E-mail Message (RFC22), PGP, POP, IMAP, HTTP, FTP
- N-Tiered Client/Server Architecture
- Universal Internet Browsing
- Multiprotocol Support

Electronic Mail

- It is one of the most widely used and popular internet applications.
- User can communicate with each other across the network
- Every user owns his own mailbox which he uses to send, receive and store messages from other users
  - Every user can be uniquely identified by his unique email address
- Mailbox principle
  - A sender does not require the receiver to be online nor the recipients to be present
  - A user's mailbox can be maintained anywhere in the internet on the server

TCP/IP Standards For Electronic Mail Service

- TCP/IP divides its mail standard into two sets.
  - One standard specifies the format for mail messages.
  - The other specifies the details of electronic mail exchange between two computers.
- What do these standards accomplish?
  - These two standards make it possible to build mail gateways that connect TCP/IP internets to some other vendor’s mail delivery system, while still using the same message format for both.
- The TCP/IP standard for mail messages specifies the exact format of mail headers; it leaves the format of the body to the user.



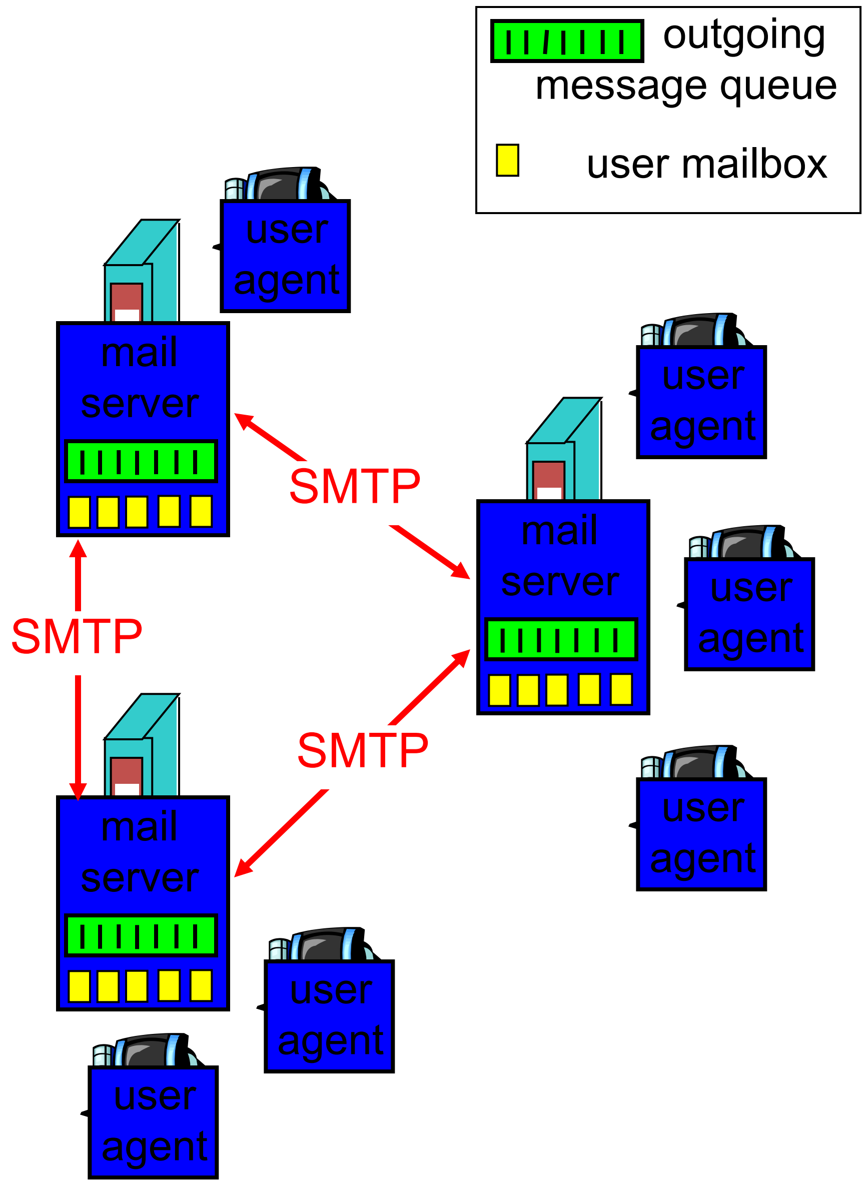
Electronic Mail Contd..

Three major components:

- user agents
- mail servers
- Simple Mail Transfer Protocol: SMTP (transfer agent)

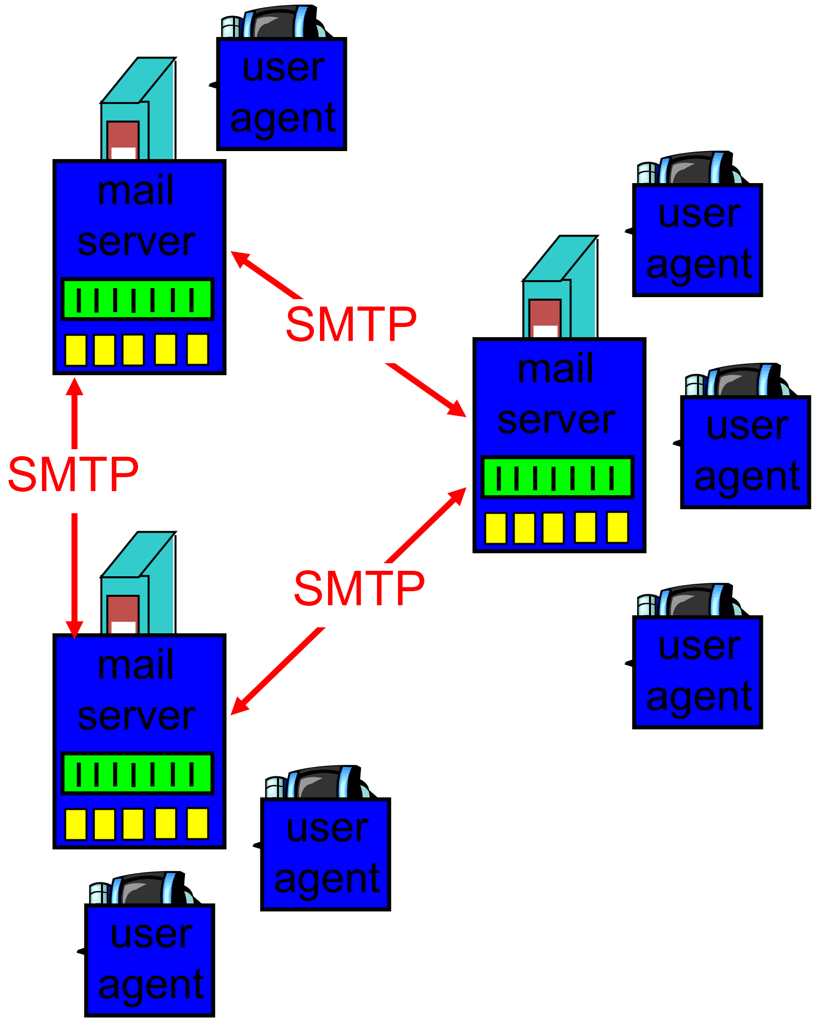
<u>User Agent</u>

- "mail reader"
- composing, editing, reading mail messages
- e.g., Eudora, Outlook, elm, Netscape Messenger
- outgoing, incoming messages stored on server



Mail Servers

- mailbox contains incoming messages for user
- message queue of outgoing (to be sent) mail messages
- SMTP protocol between mail servers to send email messages
  - client: sending mail server
  - "server": receiving mail server

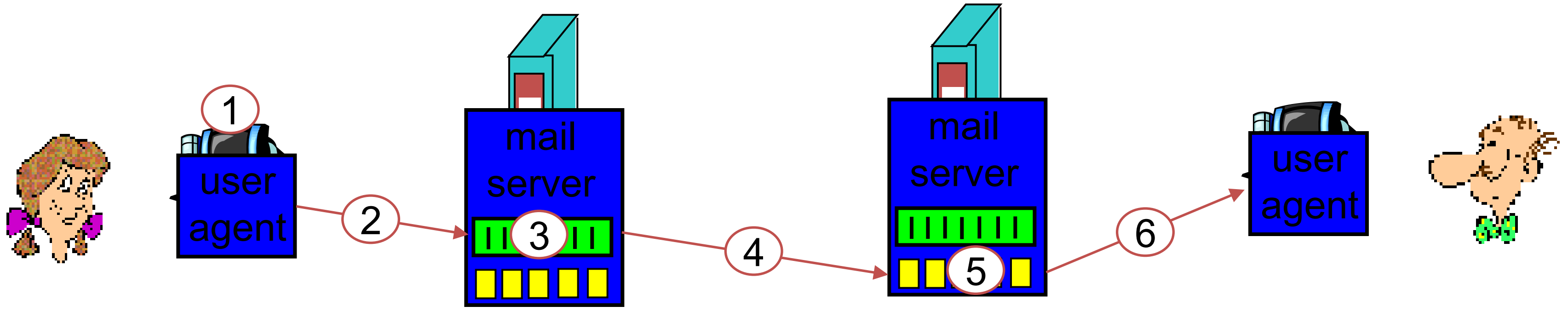


Electronic Mail: SMTP

- uses TCP to reliably transfer email message from client to server, port 25
- direct transfer: sending server to receiving server
- three phases of transfer
  - handshaking (greeting)
  - transfer of messages
  - closure
- command/response interaction
  - commands: ASCII text
  - response: status code and phrase
- messages must be in 7-bit ASCII

Scenario: Alice sends message to Bob

- Alice uses UA to compose message and send "to" bob@someschool.edu
- Alice's UA sends message to her mail server
  - message placed in message queue
- Client side of SMTP opens TCP connection with Bob's mail server
- SMTP client sends Alice's message over the TCP connection
- Bob's mail server places the message in Bob's mailbox
- Bob invokes his user agent to read message



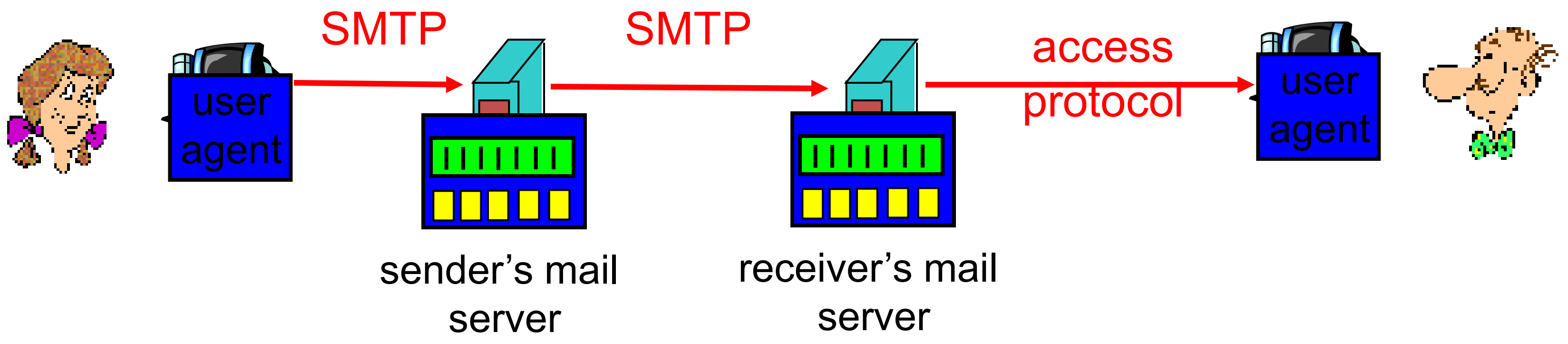
SMTP working in detail

- Communication between client and server consists of readable ASCII text.
- SMTP rigidly defines the command format
  - humans can easily read a transcript of interactions between a client and server.
- Initially a client establishes a reliable stream connection to the server and waits for the server to send a *220 READY FOR MAIL* message.
- If the server is overloaded, it may delay sending the *220* message temporarily.
- Upon receipt of *220* message, the client sends a *HELO* command.
  - The end of line marks end of command.
  - The server responds by identifying itself.

- Once communication has been established,
    - the sender can transmit one or more mail messages,
    - terminate the connection,
    - or request the server to exchange the roles of sender and receiver so messages can flow in opposite direction.
- The receiver must acknowledge each message.
    - It can also abort the entire connection or abort the current message transfer.
- Mail transactions begin with a *MAIL* command that gives sender identification as well as a *FROM:* field that contains the address to which errors should be reported.
- Response *250* means that all is well.
    - The full response consists of the text *250 OK*.

- After a successful *MAIL* command, the sender issues a series of *RCPT* commands that identify recipients of mail message.
    - The recipient must acknowledge each *RCPT* command by sending *250 OK* or by sending the error message *550 NO such user here*.
- After all *RCPT* commands have been acknowledged,
    - the sender issues a *DATA* command.
    - In essence, a *DATA* command informs the receiver that the sender is ready to transfer a complete mail message.
- The receiver responds with message 354 start mail input and specifies the sequence of characters used to terminate the mail message.

Mail access protocols



- Mail access protocol: retrieval from server
  - POP: Post Office Protocol
    - authorization (agent <-->server) and download
  - IMAP: Internet Mail Access Protocol
    - more features (more complex)
    - manipulation of stored messages on server
  - HTTP: Hotmail , Yahoo! Mail, etc.

POP

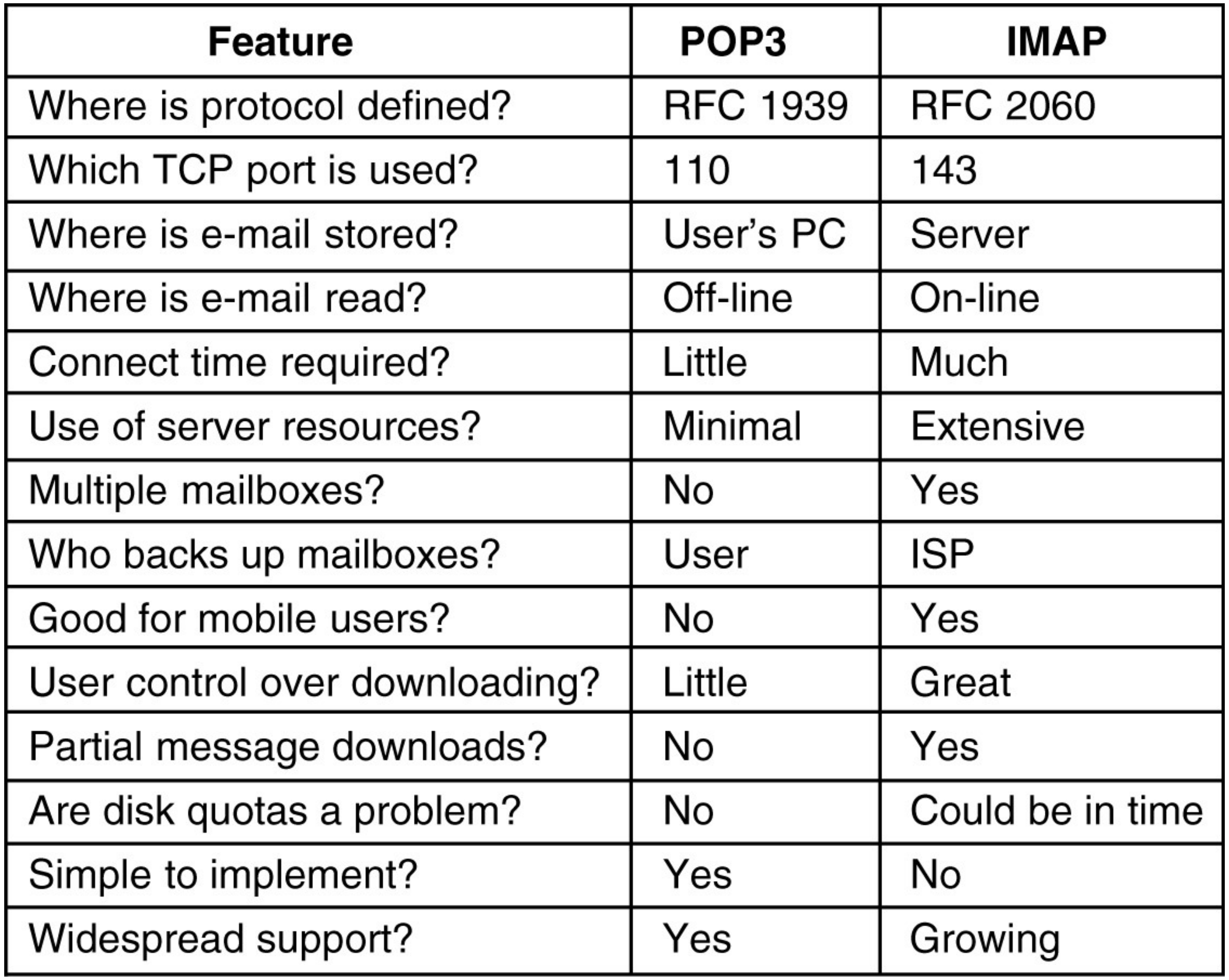
- a protocol used to retrieve e-mail from a mail server
- Most e-mail applications (sometimes called an *e-mail client*) use the POP protocol
- Examples of native mail system
  - MS outlook, Lotus Notes, MS Exchange, Eudora
- There are two versions of POP
  - The first, called *POP2,* became a standard in the mid-80's and requires SMTP to send messages.
  - The newer version, POP3, can be used with or without SMTP
  - POP3 uses TCP/IP port 110

IMAP

- It is a method of accessing electronic mail messages that are kept on a possibly shared mail server.
- In other words, it permits a "client" email program to access remote message stores as if they were local.
- For example,
  - email stored on an IMAP server can be manipulated from a desktop computer
    - at home
    - a workstation at the office
    - and a notebook computer while travelling
- IMAP uses TCP/IP port 143

POP3 vs IMAP

- With IMAP, all your mail stays on the server in multiple folders, some of which you have created.
  - This enables you to connect to any computer and see all your mail and mail folders.
  - In general, IMAP is great if you have a dedicated connection to the Internet or you like to check your mail from various locations.
- With POP3 you only have one folder, the Inbox folder.
  - When you open your mailbox, new mail is moved from the host server and saved on your computer.
  - If you want to be able to see your old mail messages, you have to go back to the computer where you last opened your mail.
- With POP3 "leave mail on server" only your email messages are on the server, but with IMAP your email folders are also on the server.



Feature	POP3	IMAP
Where is protocol defined?	RFC 1939	RFC 2060
Which TCP port is used?	110	143
Where is e-mail stored?	User’s PC	Server
Where is e-mail read?	Off-line	On-line
Connect time required?	Little	Much
Use of server resources?	Minimal	Extensive
Multiple mailboxes?	No	Yes
Who backs up mailboxes?	User	ISP
Good for mobile users?	No	Yes
User control over downloading?	Little	Great
Partial message downloads?	No	Yes
Are disk quotas a problem?	No	Could be in time
Simple to implement?	Yes	No
Widespread support?	Yes	Growing

A comparison of POP3 and IMAP

MIME

- Multipurpose Internet Mail Extensions are defined for transmission of non-ASCII data.
- MIME is a mechanism for specifying and describing the format of message bodies in a standardized way
- It allows ordinary data to be encoded in ASCII and transmitted in standard email.
- MIME header specifies:
  - Version of MIME used.
  - Type of data being sent.
  - Encoding used to convert data to ASCII.

MIME Multipart Messages

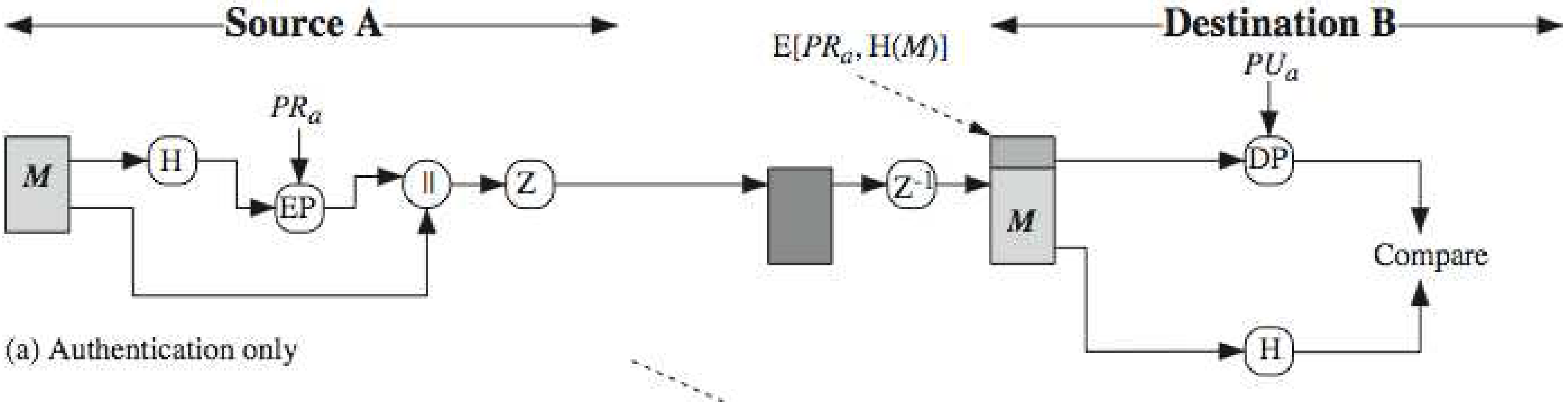
- Four subtypes for multipart messages:
- Mixed
  - Allows single message to contain multiple type messages.
  - Possible to send text, graphics and sound in single message.
- Alternative
  - Allows single message to include multiple representations of same data.
  - Useful when sending same message to multiple recipients.
- Parallel
  - Permits message to include subparts that should be viewed together(Like video and audio).
- Digest
  - Permits a message to contain set of other messages.
  - E.g., collection of e-mail messages from a discussion.

Securing e-mail (PGP)

- PGP -> Pretty Good Privacy
- General purpose application to protect (encrypt and/or sign) files
- Can be used to protect e-mail messages
- Can be used by corporations as well as individuals
- Based on strong cryptographic algorithms (IDEA, RSA, SHA-1)
- Available free of charge at http://www.pgpi.org
- First version developed by Phil Zimmermann
- PGP is now on an Internet standards track (RFC 3156)

PGP Operation: Authentication

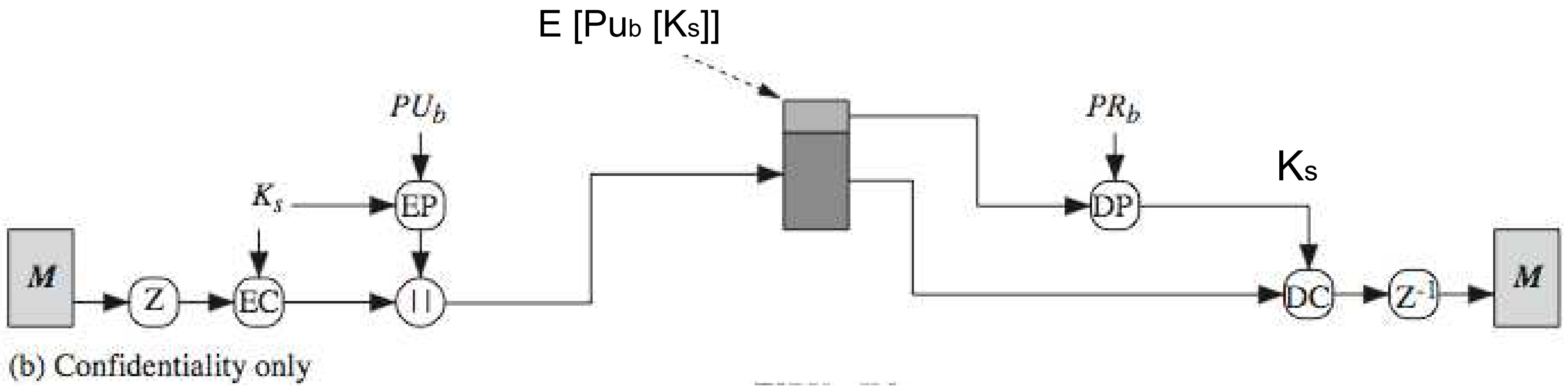
- Sender creates message
- Make SHA-1 160-bit hash of message
- Attached RSA signed hash to message
- Receiver decrypts & recovers hash code
- Receiver verifies received message hash
    - if match, message is accepted as authentic



(a) Authentication only

PGP Operation: Confidentiality

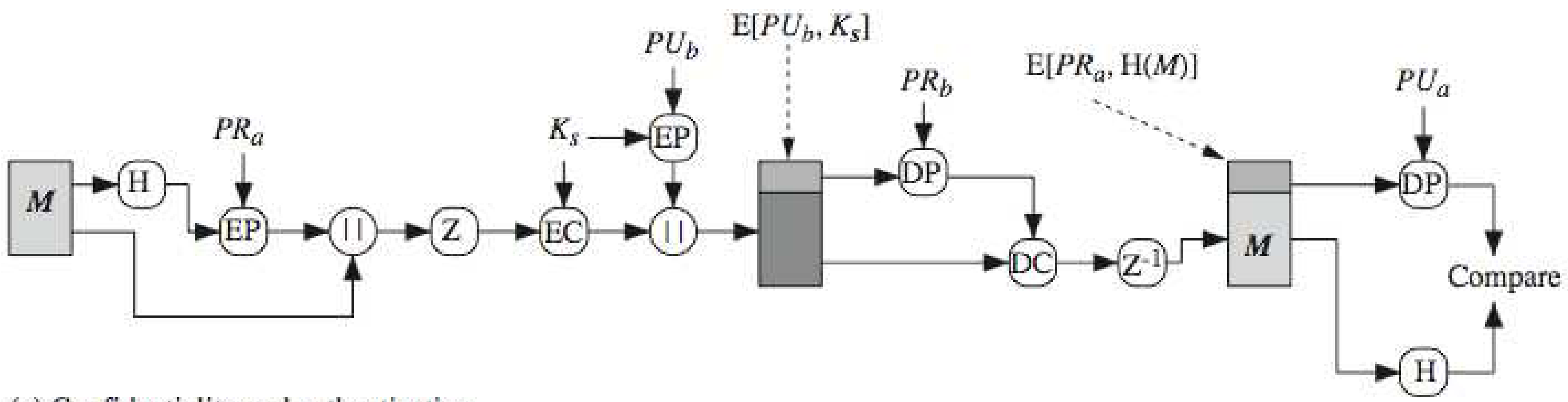
- Sender forms 128-bit random session key
- Encrypts message with session key
- Attaches session key encrypted with RSA
- Receiver decrypts & recovers session key
- Session key is used to decrypt message



(b) Confidentiality only

PGP Operation: Confidentiality + Authentication

- Can use both services on same message
- Create signature & attach to message
- Encrypt both message & signature
- Attach RSA encrypted session key



(c) Confidentiality and authentication

PGP Operation : Compression

- By default PGP compresses message after signing but before encrypting
  - so can store uncompressed message & signature for later verification
- Uses ZIP compression algorithm

PGP Operation : Email Compatibility

- PGP will have binary data to send (encrypted message etc)
- However email was designed only for text
- Hence PGP must encode raw binary data into printable ASCII characters
- Uses radix-64 algorithm
  - maps 3 bytes to 4 printable chars
  - also appends a CRC

PGP Operation : Segmentation

- Another constraint of e-mail is that there is usually a maximum message length.
- PGP automatically segments an encrypted message into blocks of an appropriate length.
- On receipt, the segments must be re-assembled before the decryption process

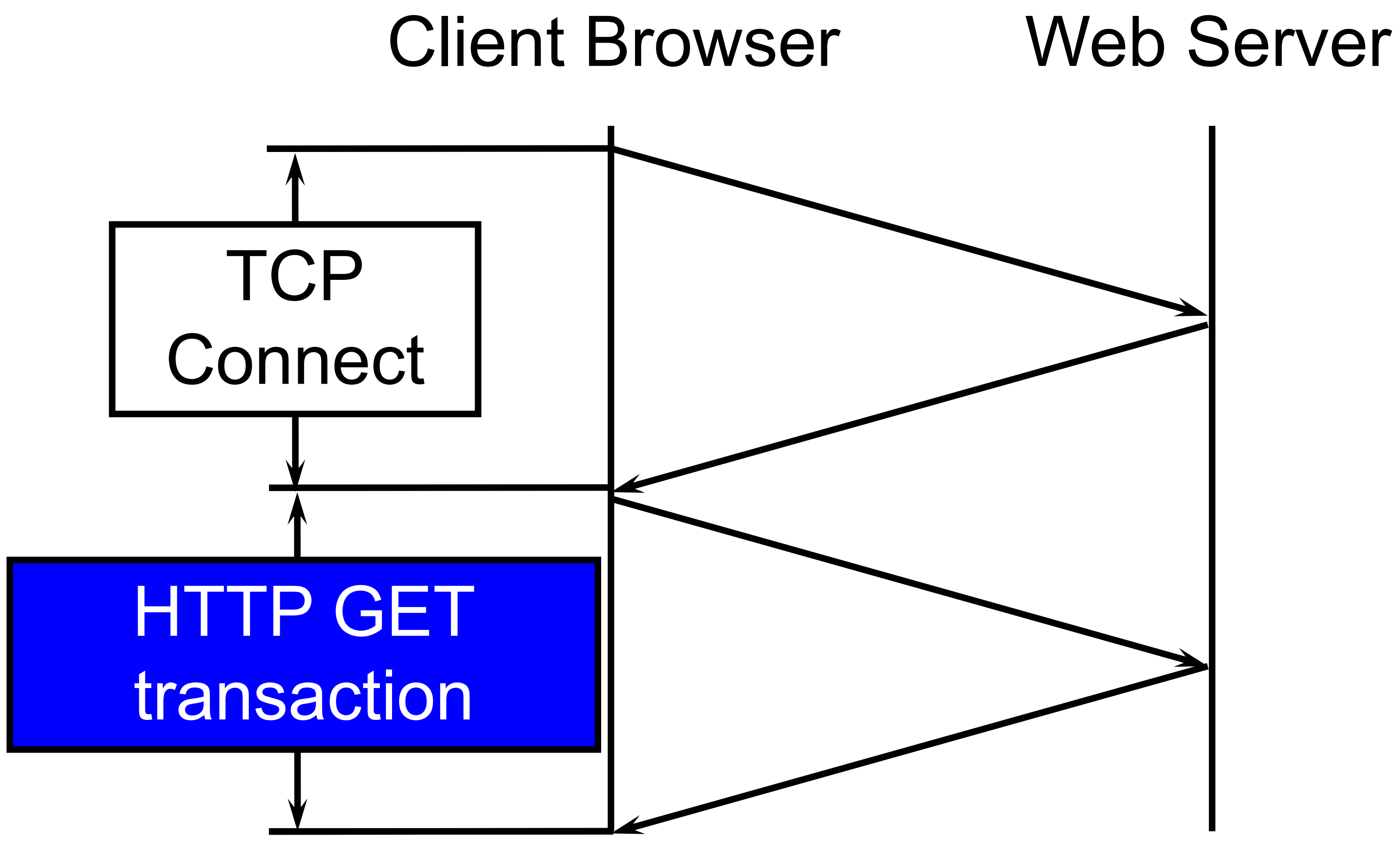
HTTP and HTTPS

- HTTP stands for Hyper Text Transport Protocol
- HTTP is an application-level protocol for distributed, collaborative, hypermedia information systems.
- This is the foundation for data communication for the World Wide Web (ie. internet) since 1990
  - allows the transmitting and receiving of information across the Internet
- The S stands for "Secure“ in HTTPS i.e., Secure hypertext transfer protocol
  - HTTPS is http using a Secure Socket Layer (SSL).

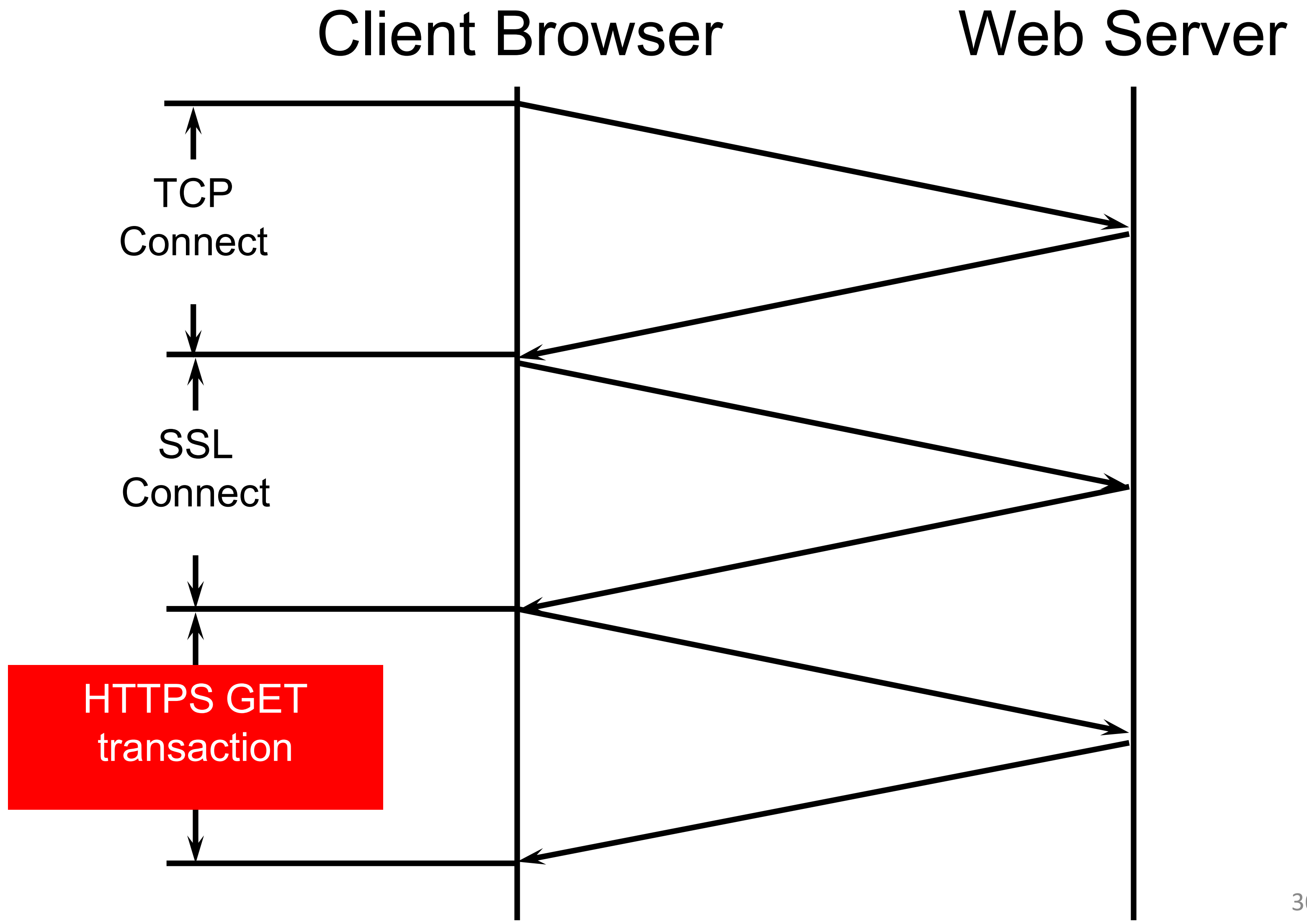
- Secure HTTP (HTTPS) is one of the popular protocols to transfer sensitive data over the Internet.
    - A secure socket layer is an encryption protocol invoked on a Web server that uses HTTPS.
    - Most implementations of the HTTPS protocol involve online purchasing or the exchange of private information.
- HTTPS is only slightly slower than HTTP
- Why is HTTPS not used for all web traffic?
    - Slows down web servers
    - Breaks Internet caching
        - ISPs cannot cache HTTPS traffic
        - Results in increased traffic at web site

- HTTP is stateless
  - The server and client are aware of each other only during a current request.
  - Afterwards, both of them forget about each other.
  - Due to this nature of the protocol, neither the client nor the browser can retain information between different request across the web pages.
- Protocol that maintain states are complex
- Cookies maintain states

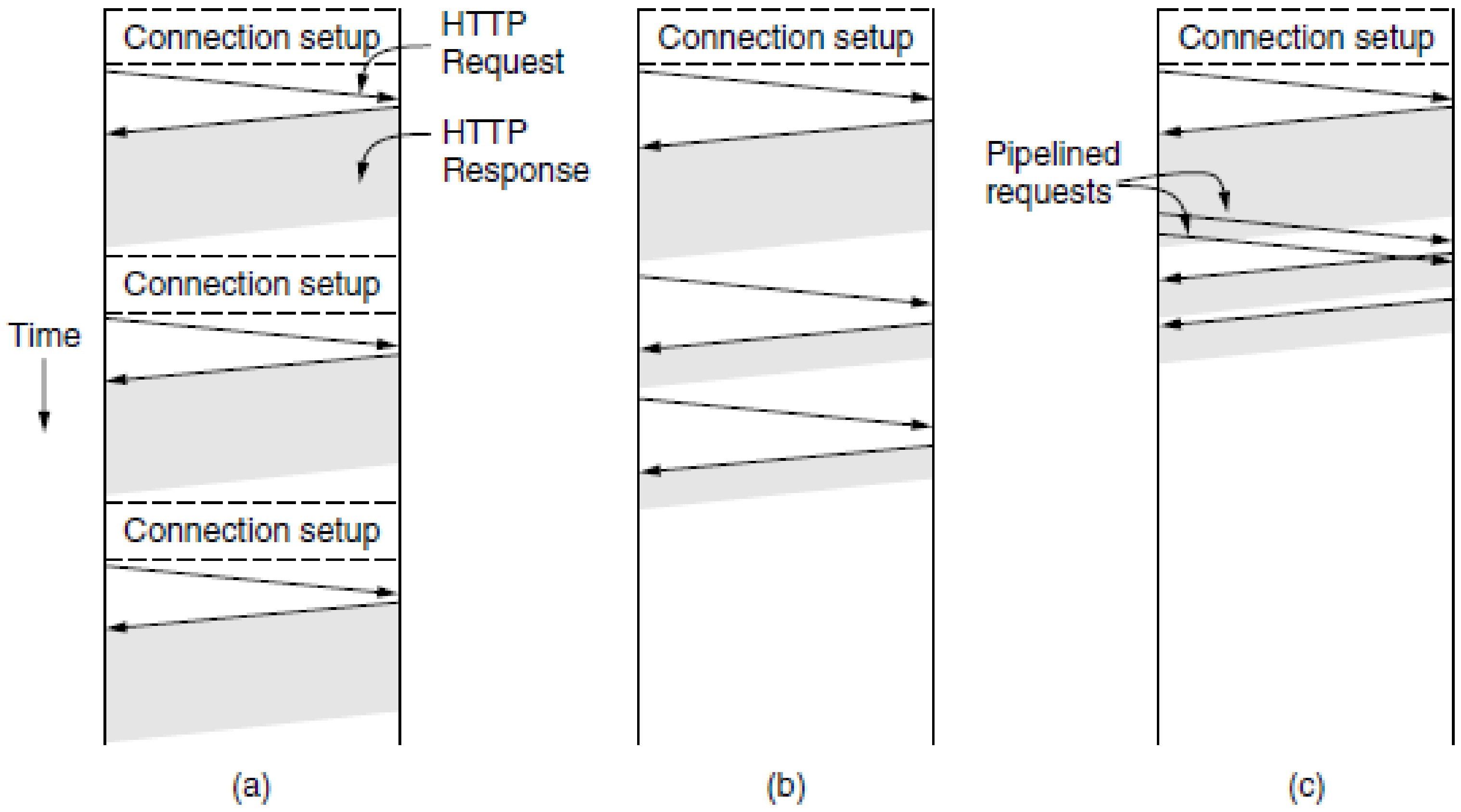
HTTP Transaction



HTTPS Transaction



HTTP Connection



HTTP with (a) multiple connections and sequential requests. (b) A persistent connection and sequential requests. (c) A persistent connection and pipelined requests.

- Figure (a) shows non-persistent connections
  - Browser makes connection and request for each object
  - Server parses request, responds and closes connection
  - Performance problem
- Figure (b) shows persistent connections
  - On same TCP connection server parses request, responds, parses new request upon responds and so on
  - Improves performance
- Figure (c) shows pipelining with persistent connections
  - Send next request before previous response is received
  - Pipelining with persistent connection improves performance

Overall operation of HTTP

- The HTTP protocol is a request/response protocol.
- Client
    - A client sends a request to the server in the form of
        - a request method
        - URI, and
        - protocol version
        - followed by a MIME-like message containing
            - request modifiers
            - client information, and
            - possible body content over a connection with a server.

- Server
  - The server responds with
    - ➢a status line
    - ➢the message's protocol version and
    - ➢a success or error code,
    - ➢followed by a MIME-like message containing
      - server information,
      - entity meta information, and
      - possible entity-body content.

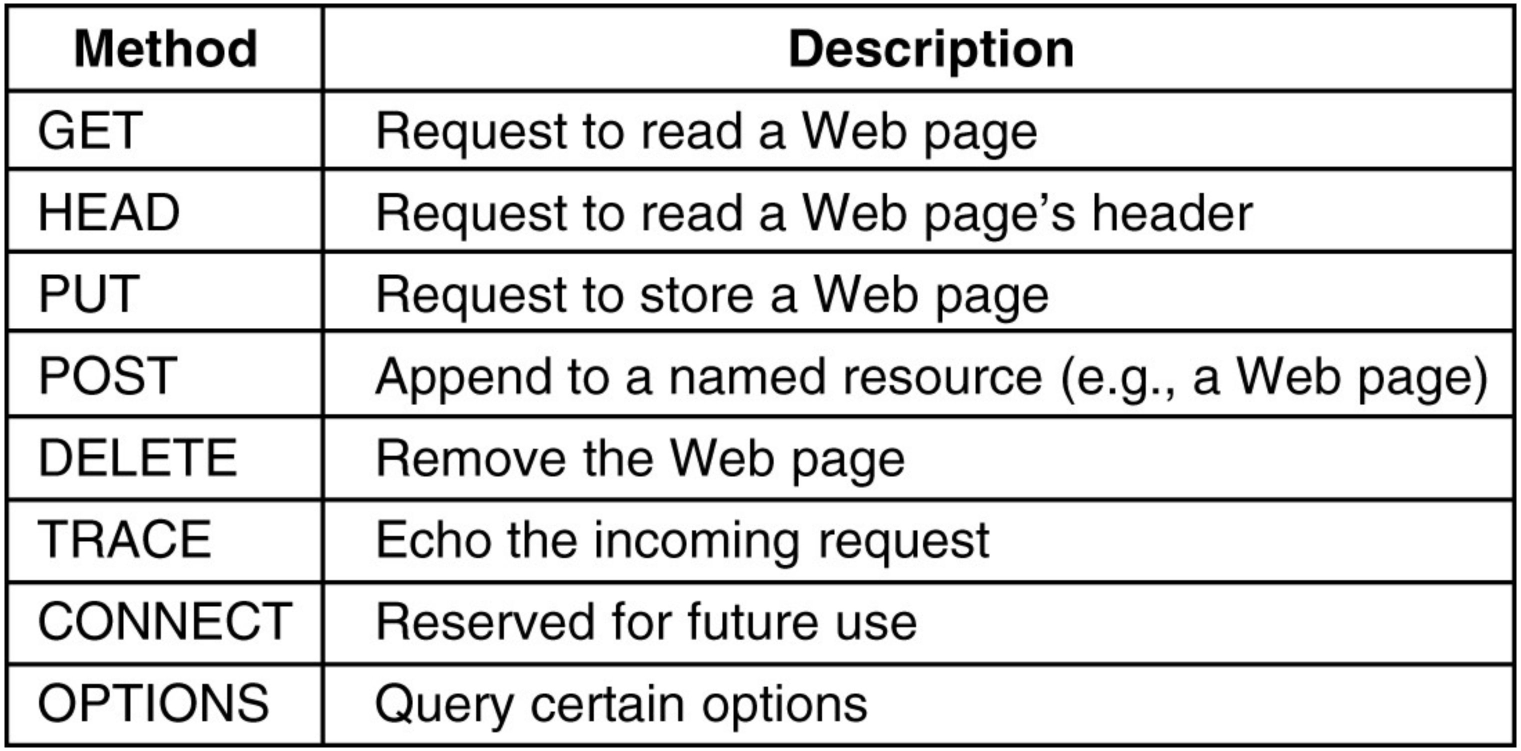
HTTP Version

- HTTP uses a <major>.<minor> numbering scheme to indicate versions of the protocol.
- The version of an HTTP message is indicated by an HTTP-Version field in the first line.
- Here is the general syntax of specifying HTTP version number
  - HTTP-Version = "HTTP" "/" 1*DIGIT "." 1*DIGIT
  - Example
    - HTTP/1.0
    - HTTP/1.1

URI (Uniform Resource Identifiers)

- URI is simply formatted, case-insensitive string containing name, location etc to identify a resource
  - for example a website, a web service etc.
- A general syntax of URI used for HTTP is as follows:
  - URI = "http:" "//" host [":" port] [abs_path ["?" query]]
- Here if the port is empty or not given,
  - port 80 is assumed for HTTP and
  - an empty abs_path is equivalent to an abs_path of "/“
- For example, the following three URIs are equivalent:
  - http://abc.com:80/~smith/home.html
  - http://ABC.com/%7Esmith/home.html
  - http://ABC.com:/%7esmith/home.html

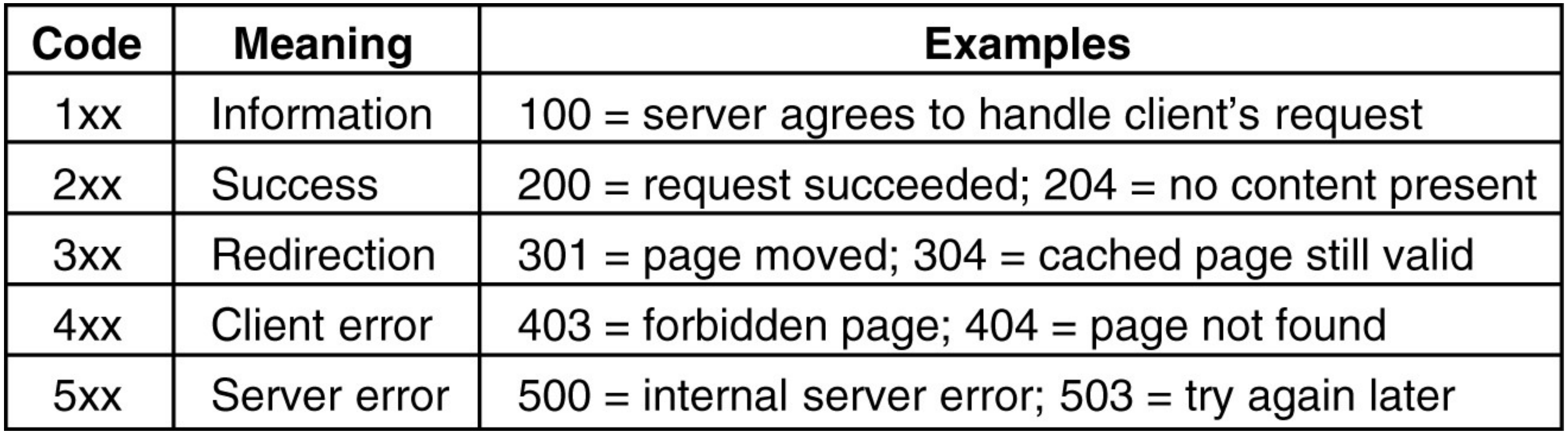
HTTP Methods



Method	Description
GET	Request to read a Web page
HEAD	Request to read a Web page’s header
PUT	Request to store a Web page
POST	Append to a named resource (e.g., a Web page)
DELETE	Remove the Web page
TRACE	Echo the incoming request
CONNECT	Reserved for future use
OPTIONS	Query certain options

The built-in HTTP request methods

HTTP Methods Contd..



Code	Meaning	Examples
1xx	Information	100 = server agrees to handle client's request
2xx	Success	200 = request succeeded; 204 = no content present
3xx	Redirection	301 = page moved; 304 = cached page still valid
4xx	Client error	403 = forbidden page; 404 = page not found
5xx	Server error	500 = internal server error; 503 = try again later

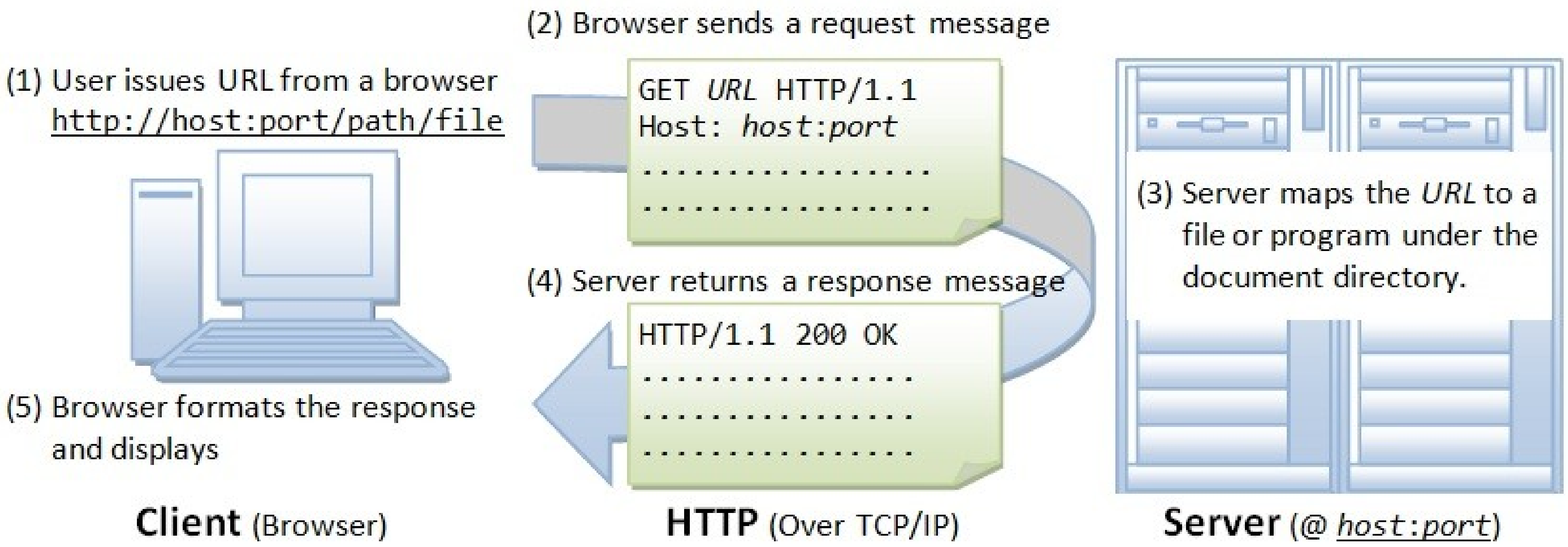
The status code response groups

HTTP Message Headers

Header	Type	Contents
User-Agent	Request	Information about the browser and its platform
Accept	Request	The type of pages the client can handle
Accept-Charset	Request	The character sets that are acceptable to the client
Accept-Encoding	Request	The page encodings the client can handle
Accept-Language	Request	The natural languages the client can handle
Host	Request	The server's DNS name
Authorization	Request	A list of the client's credentials
Cookie	Request	Sends a previously set cookie back to the server
Date	Both	Date and time the message was sent
Upgrade	Both	The protocol the sender wants to switch to
Server	Response	Information about the server
Content-Encoding	Response	How the content is encoded (e.g., gzip)
Content-Language	Response	The natural language used in the page
Content-Length	Response	The page's length in bytes
Content-Type	Response	The page's MIME type
Last-Modified	Response	Time and date the page was last changed
Location	Response	A command to the client to send its request elsewhere
Accept-Ranges	Response	The server will accept byte range requests
Set-Cookie	Response	The server wants the client to save a cookie

Some HTTP message headers

HTTP Example

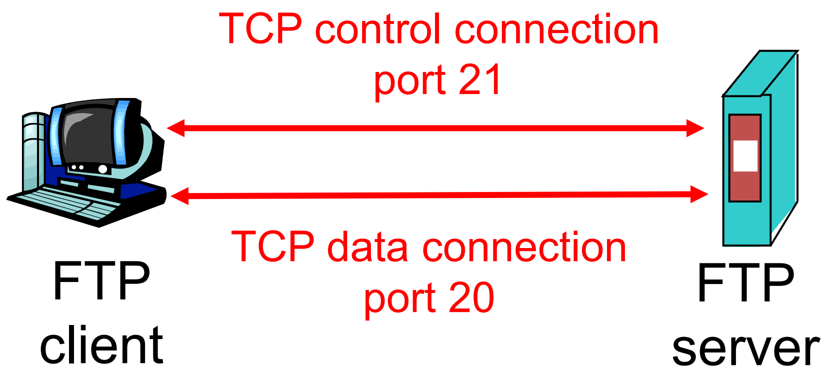

(1) User issues URL from a browser
http://host:port/path/file
(2) Browser sends a request message
GET URL HTTP/1.1
Host: host:port
(3) Server maps the URL to a file or program under the document directory.
(4) Server returns a response message
HTTP/1.1 200 OK
(5) Browser formats the response and displays
Client (Browser)
HTTP (Over TCP/IP)
Server (@ host:port)

File Transfer Protocol : FTP

- Allows a user to copy files to/from remote hosts
- FTP uses the services of TCP
- It needs two TCP connections
  - The well-known port 21 is used for the control connection
  - and the well-known port 20 for the data connection
- Goal of FTP (Objective of FTP)
  - promote sharing of files
  - encourage indirect use of remote computers
  - shield user from variations in file storage
  - transfer data reliably and efficiently

FTP: Working
separate control, data connections

- FTP client contacts FTP server at port 21, specifying TCP as transport protocol
- Client obtains authorization over control connection
- Client browses remote directory by sending commands over control connection.
- When server receives a command for a file transfer, the server opens a TCP data connection to client
- After transferring one file, server closes connection.



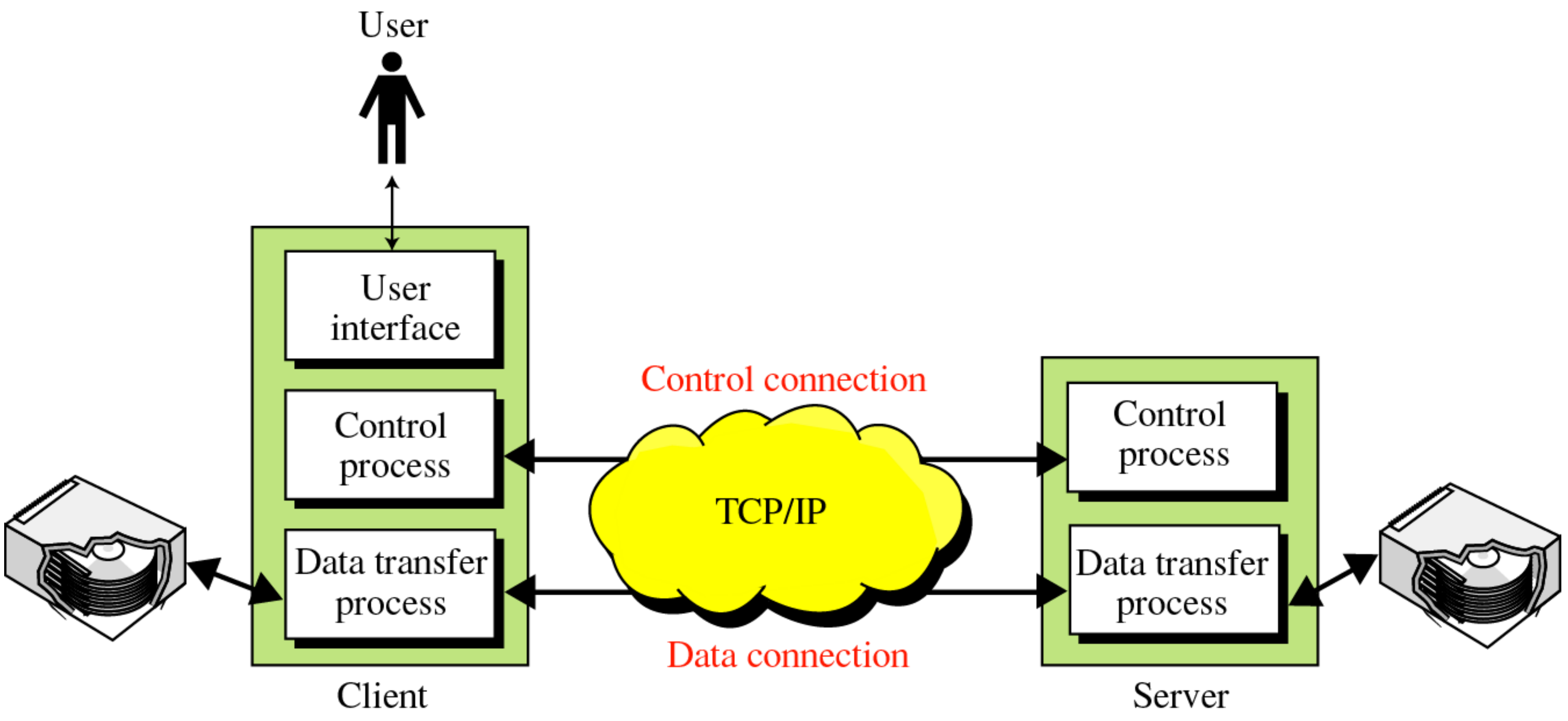
- Server opens a second TCP data connection to transfer another file.
- Control connection: "out of band"
- FTP server maintains "state": current directory, earlier authentication

FTP Replies

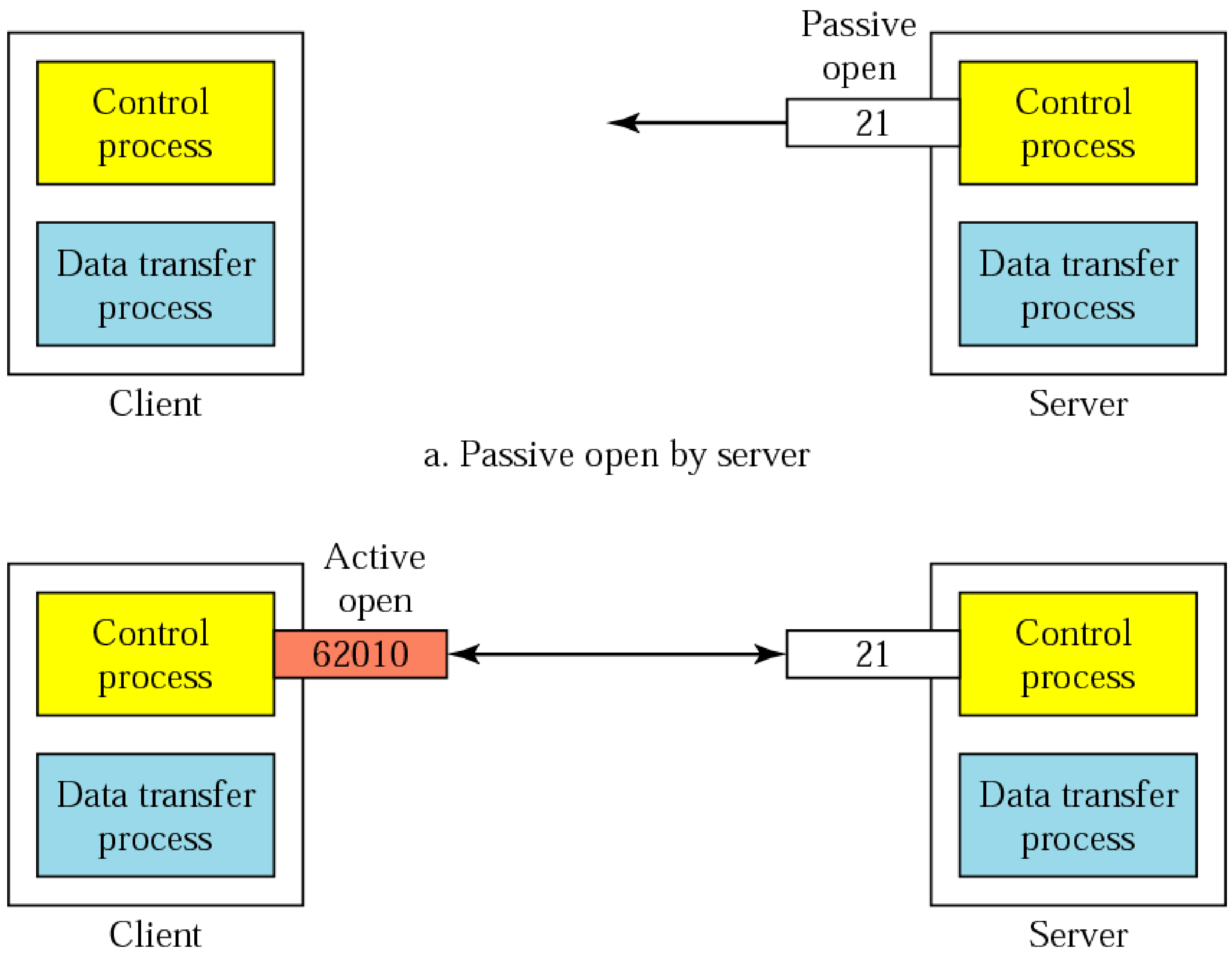
- All replies are sent over control connection.
- Replies are a single line containing
  - 3 digit status code (sent as 3 numeric chars).
  - text message.
- FTP Reply Status Code
  - First digit of status code indicates type of reply:
    - 1: Positive Preliminary Reply (got it, but wait).
    - 2: Positive Completion Reply (success).
    - 3: Positive Intermediate Reply (waiting for more information).
    - 4: Transient Negative Completion (error - try again).
    - 5: Permanent Negative Reply (error - can’t do)

- 2nd digit indicates function groupings.
- 3rd digit indicates specific problem within function group.

FTP Model


User
User interface
Control process
Data transfer process
Client
Control connection
TCP/IP
Data connection
Control process
Data transfer process
Server

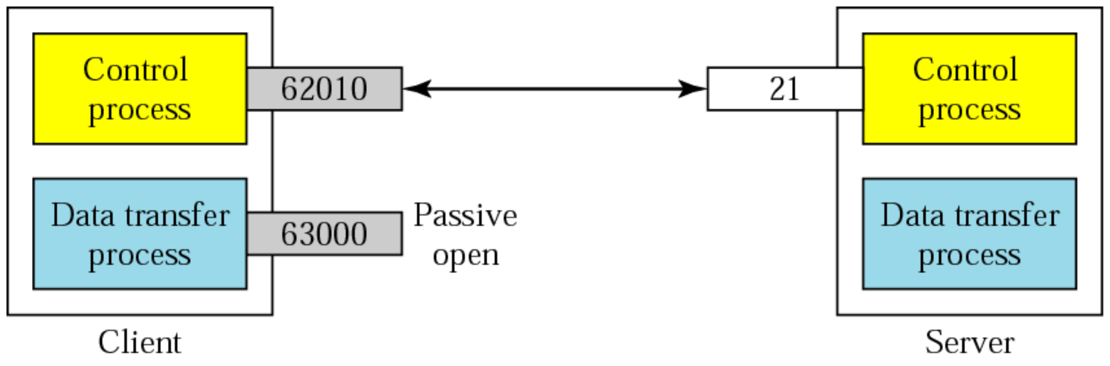
FTP Connections: The control connection



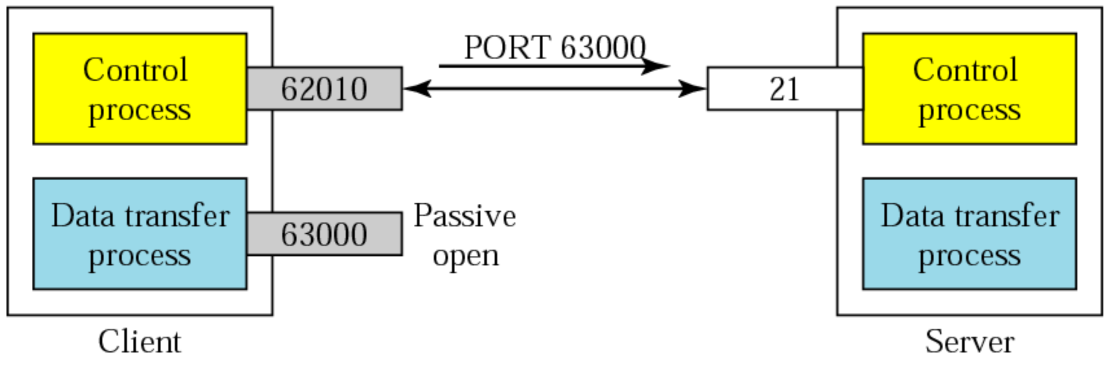
a. Passive open by server

b. Active open by client

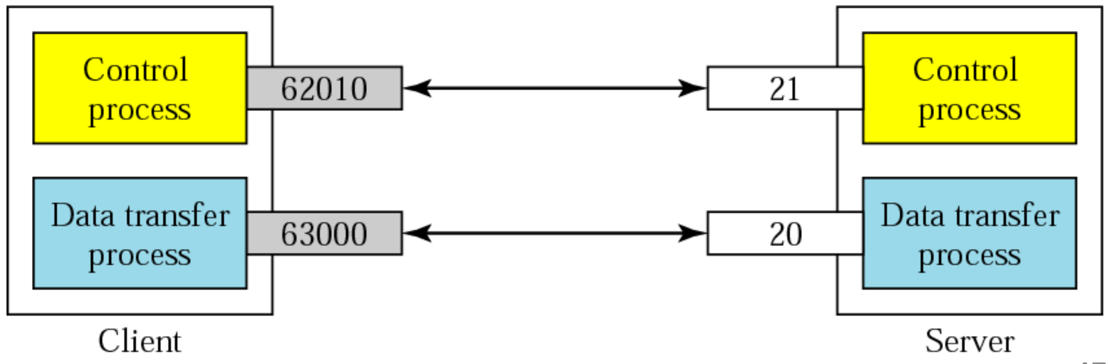
FTP : The data connection



a. Passive open by client



b. Sending ephemeral port number to server

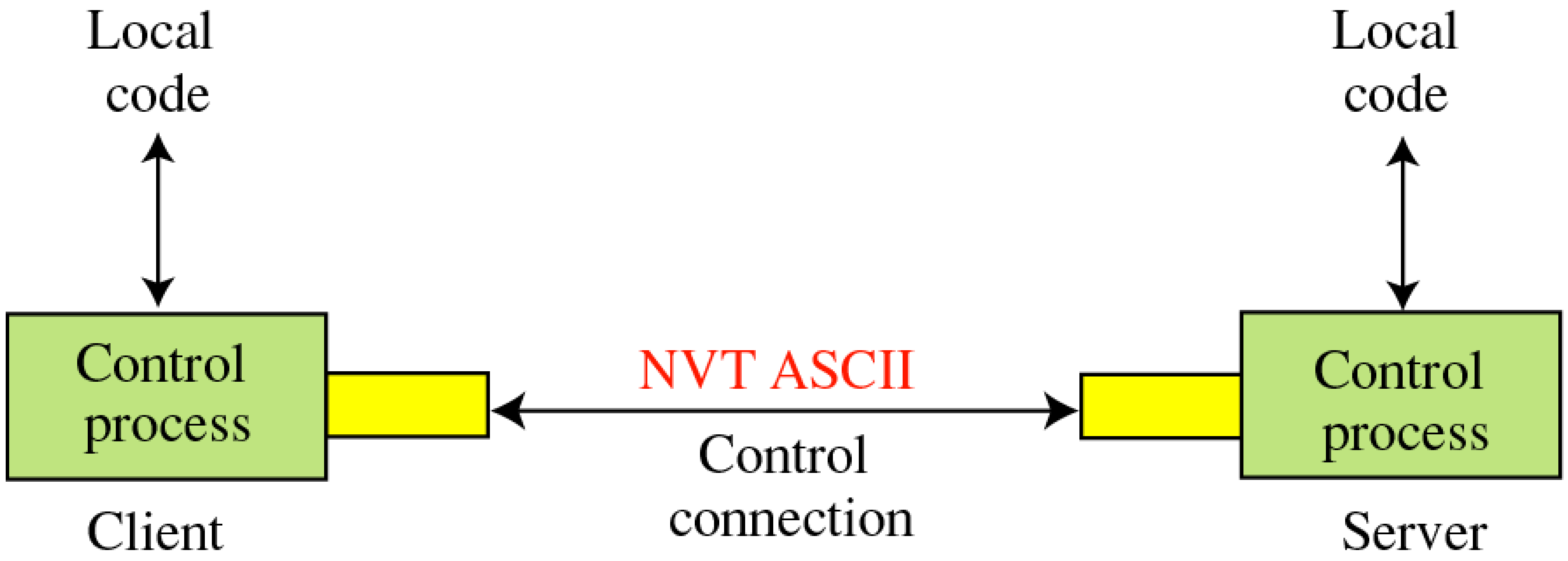


c. Active open by server

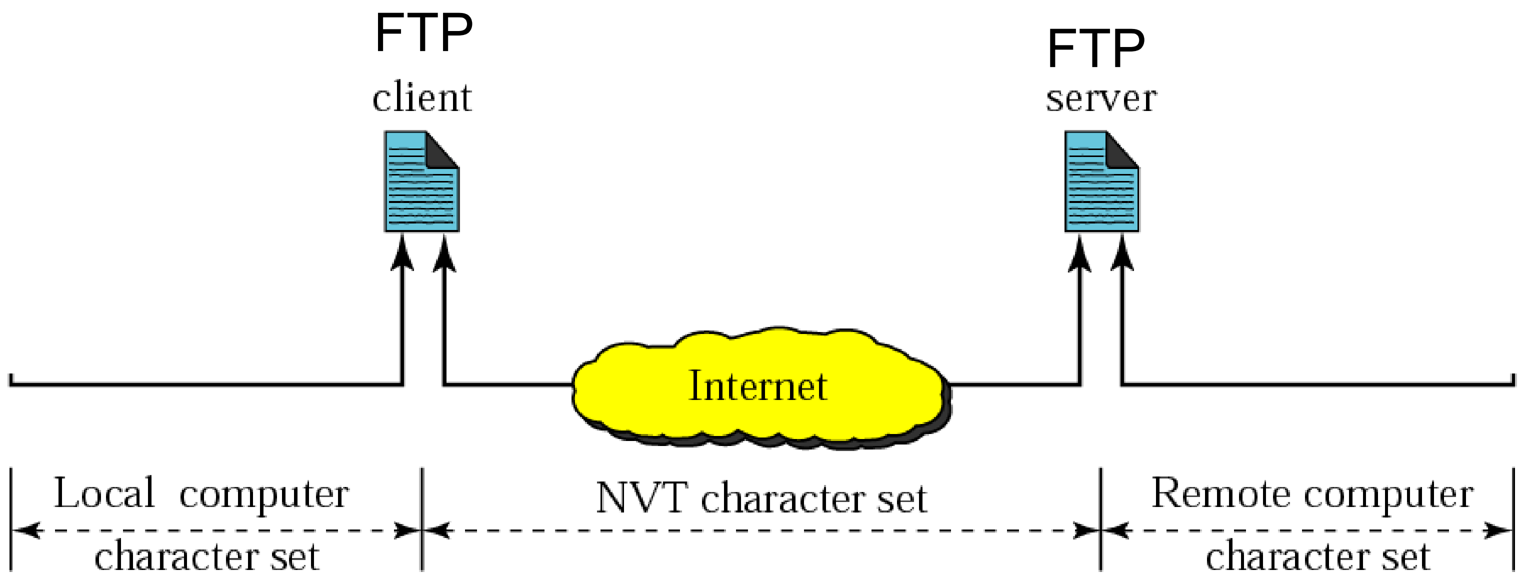
The Data Connection

- Uses Server’s well-known port 20
  - Client issues a passive open on an ephemeral port, say *x*.
  - Client uses PORT command to tell the server about the port number *x*.
  - Server issues an active open from port 20 to port *x*.
  - Server creates a child server/ephemeral port number to serve the client

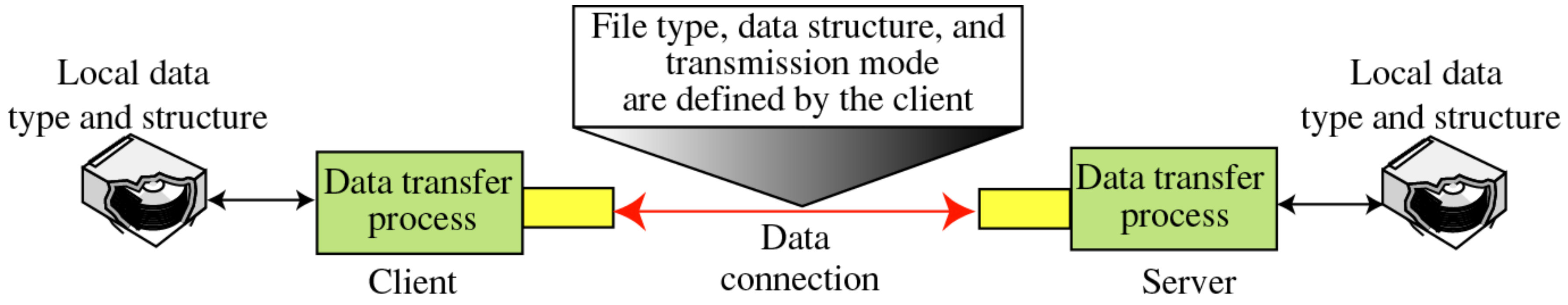
Communication using the control connection



NVT


FTP
client
FTP
server
Internet
Local computer
character set
NVT character set
Remote computer
character set

Using the data connection


File type, data structure, and transmission mode are defined by the client
Local data type and structure
Data transfer process
Client
Data connection
Data transfer process
Server
Local data type and structure

Transmission Modes

- Independent of data type and file structure

Stream	S	file is transmitted as stream of bytes
Block	B	file sent as sequence of blocks preceded by header information ; allows restart of an interrupted transfer
Compressed	C	data compressed using simple compression techniques eg., run length encoding

File Structures

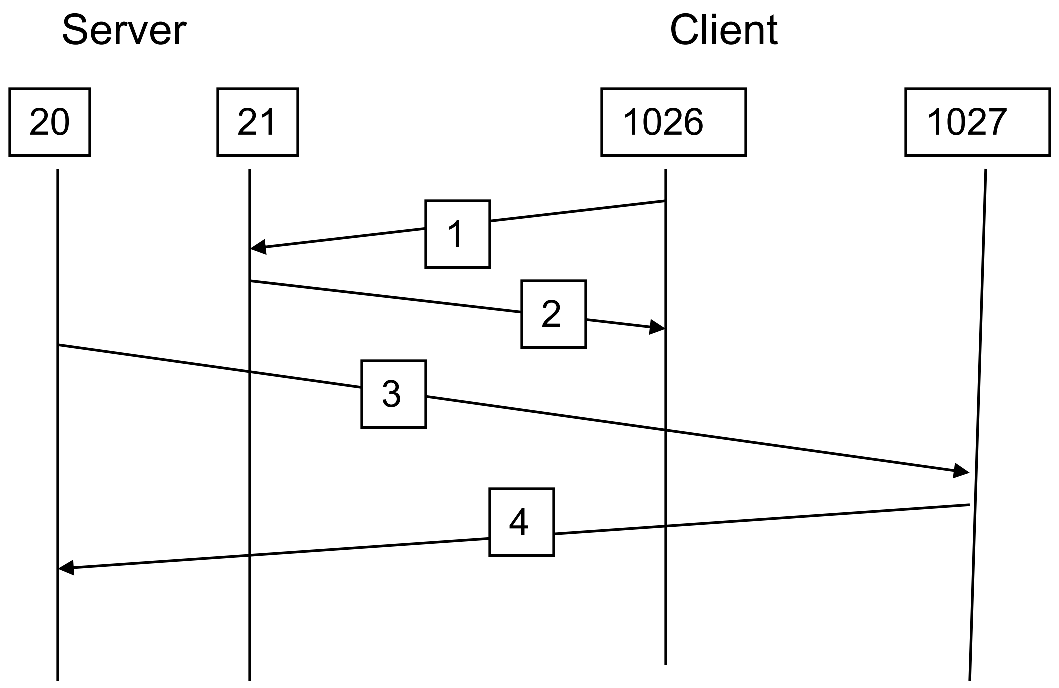
- Operating System store files in different structures
- FTP defined file structures for transporting files

File	F	Unstructured, sequence of bytes
Record	R	Series of records
Page	P	Series of data blocks (pages)

- Default file structure is File (F)
- File Structure specified using STRU command

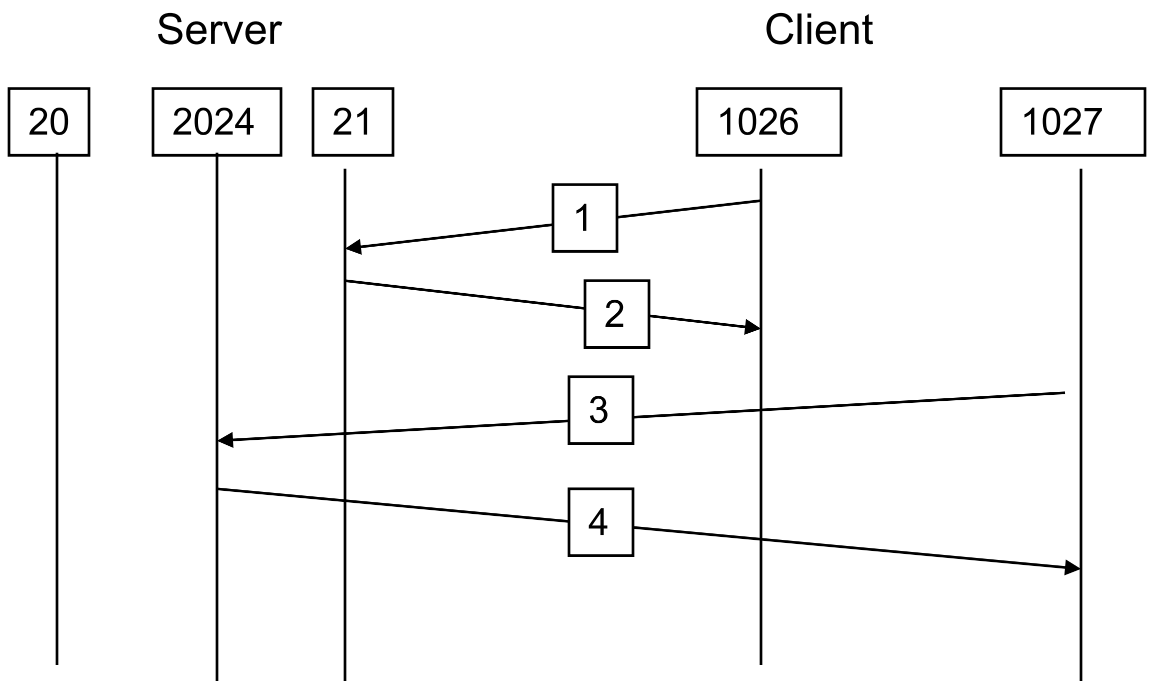
Active Mode FTP

- Client connect from a random unprivileged port (n > 1023) to the servers command port (21) and sends port command to tell server to connect to n+1 then listens on the next higher unprivileged port (n+1) for server responses. The server connects from it’s data port (20) to the client data port (n+1)



Passive Mode FTP

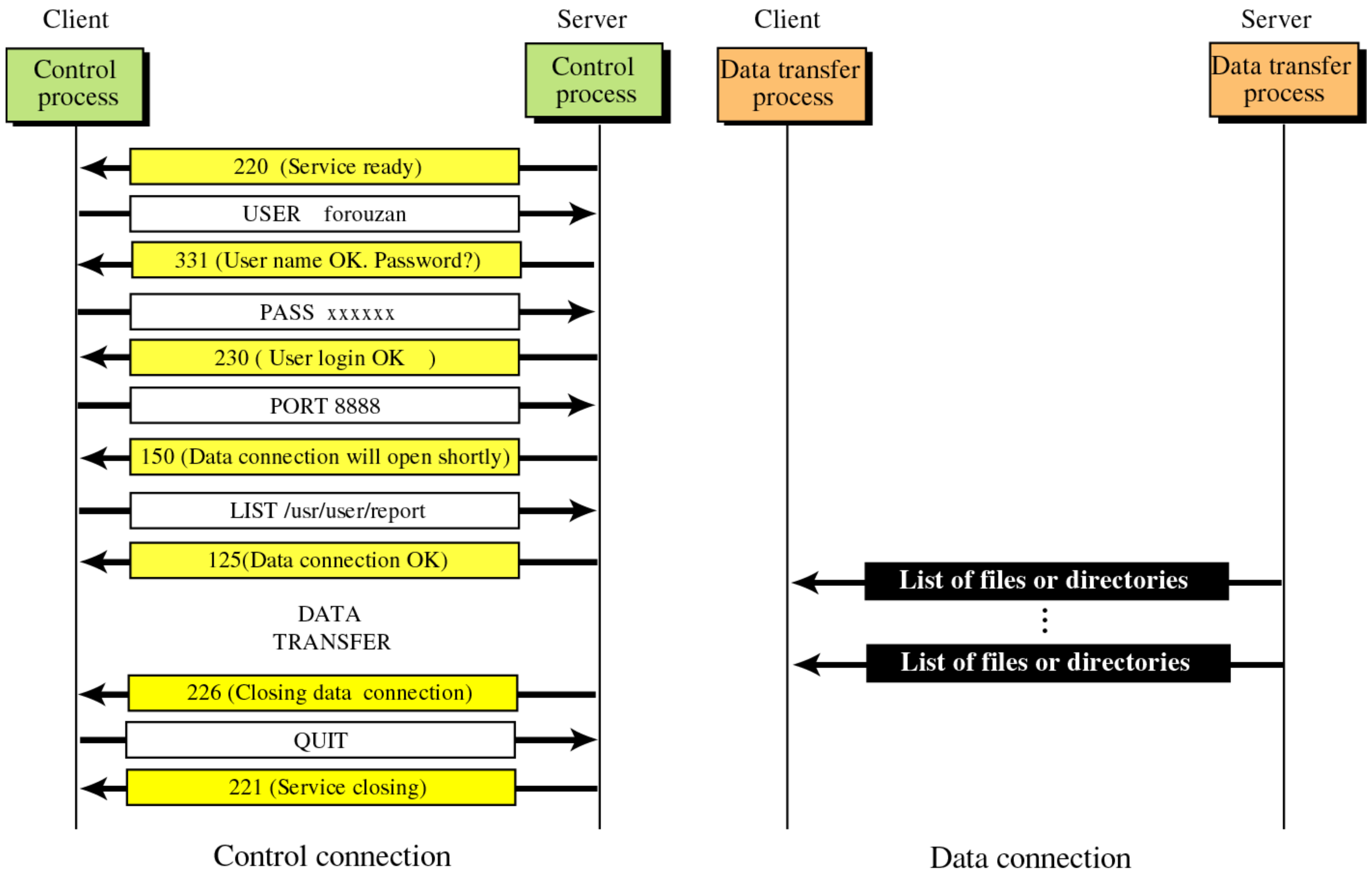
- Client opens two random unprivileged ports (n > 1023 and n+1; eg., 1026 and 1027) and connects the first port (n) to server command port 21 and issues a pasv command (server sends port to use for data); client connects to servers specified data port, server completes connection.



Anonymous FTP

- These accounts have limited access rights, as well as some operating restrictions
- Internet users log in with username anonymous and a password with their email address
- Using email addresses allows the administrators to monitor who is using their services
- To retrieve a file, users need to know the host to connect to and the pathname of the file
- Note that there are some variations on how users connect and use specific hosts, i.e. don’t assume all are set up the same
- There are differences in the implementation of FTP commands at sites

Example 1


Client
Control process
Server
Control process
Client
Data transfer process
Server
Data transfer process
220 (Service ready)
USER forouzan
331 (User name OK. Password?)
PASS xxxxxx
230 (User login OK)
PORT 8888
150 (Data connection will open shortly)
LIST /usr/user/report
125(Data connection OK)
DATA TRANSFER
List of files or directories
List of files or directories
226 (Closing data connection)
QUIT
221 (Service closing)
Control connection
Data connection

Trivial File Transfer Protocol (TFTP)

- UDP port 69
- Simple protocol, usually used to transfer configuration files
- Usually used for transferring boot file for diskless hosts (X-Stations) or updating NVRAM
- Typically used in short distance, low noise environments
- Server is usually implement in firmware for updating things like routers, bios...
- Because of its compact size:
  - no error recovery like TCP based FTP
  - no command structure like FTP
  - cannot list directories
  - transfers to server are to a single configured directory

Traditional Host Systems

A Central Processing System (Mainframe) provides all processing. Local Terminals are responsible for display and keyboard for user input and viewing capabilities. Local Terminals do not contain any intelligent processing capabilities.

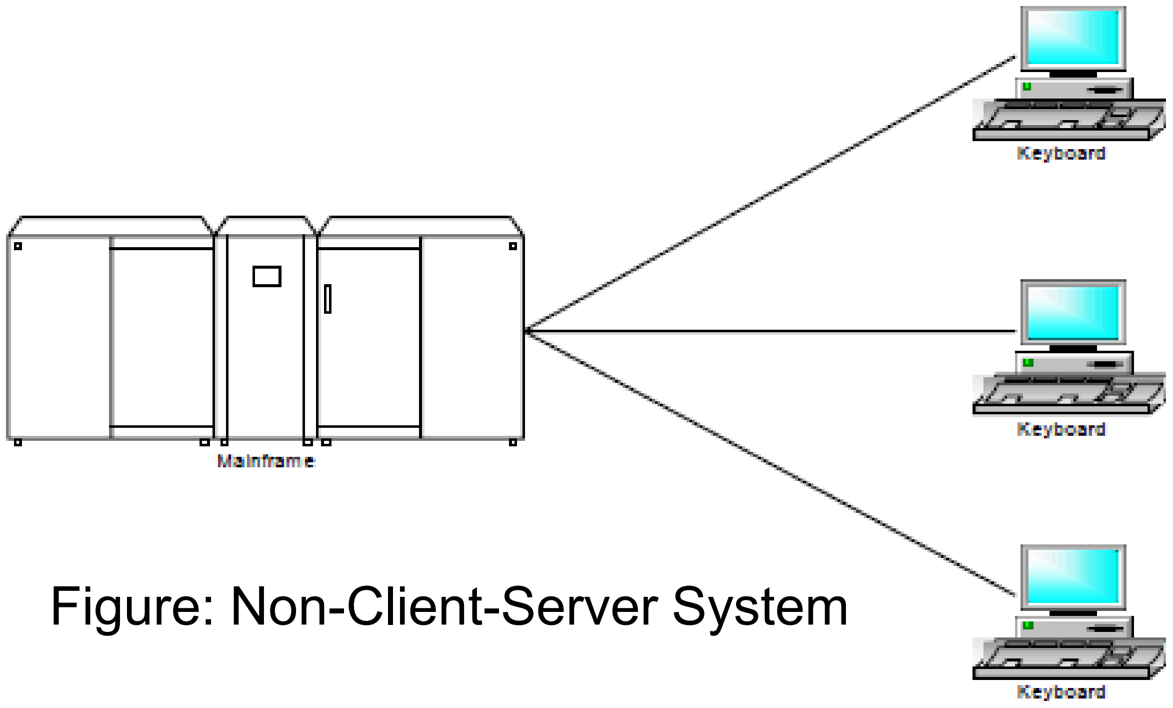


Figure: Non-Client-Server System

File Server and retrieval processing provided by File Server. Word Processing and spreadsheet processing provided by PC workstation.

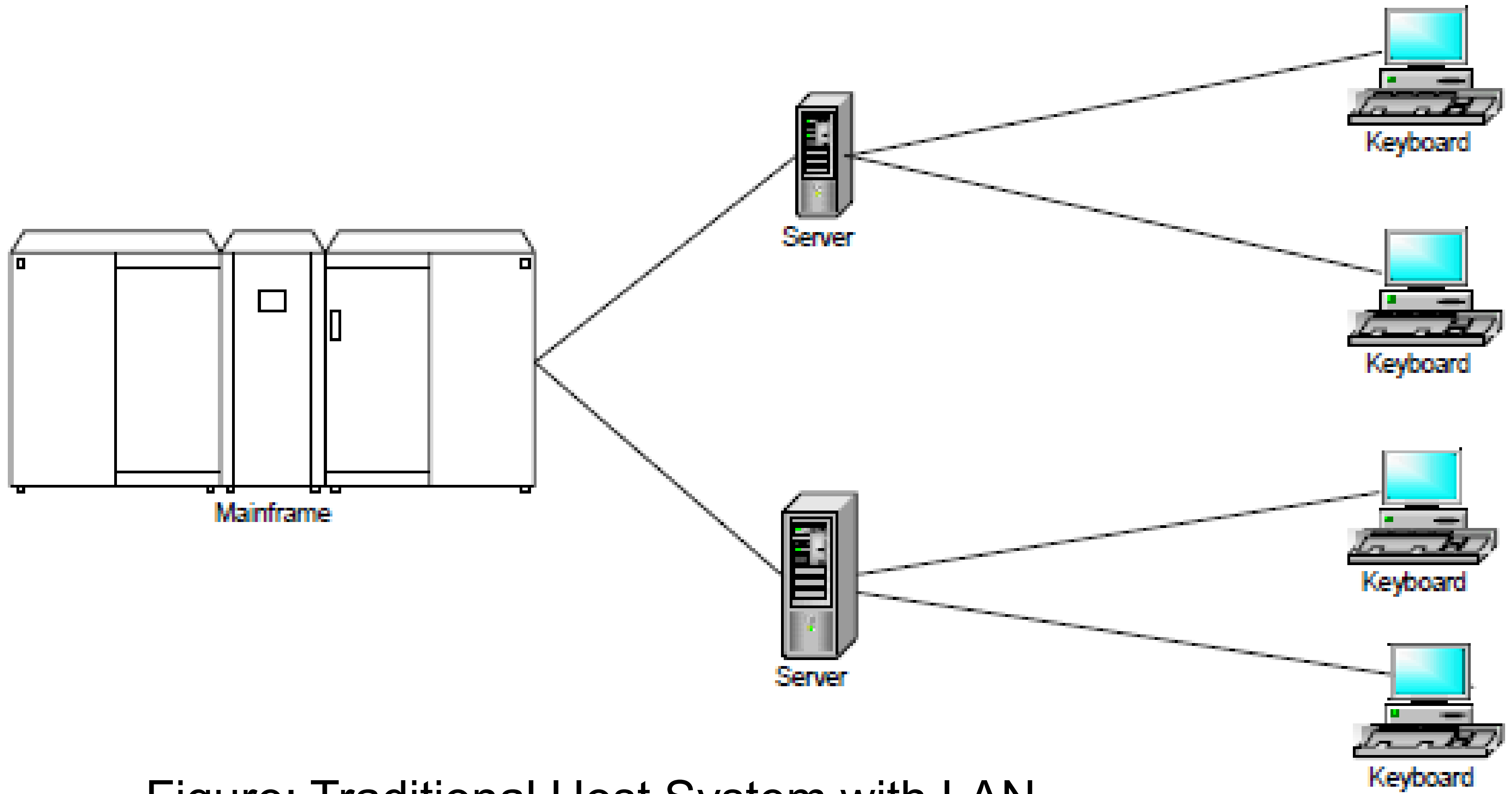


Figure: Traditional Host System with LAN

Distributed Systems

Distributed System
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.

Replication
Ensures data at all sites in a distributed system reflects any changes made anywhere in the system.

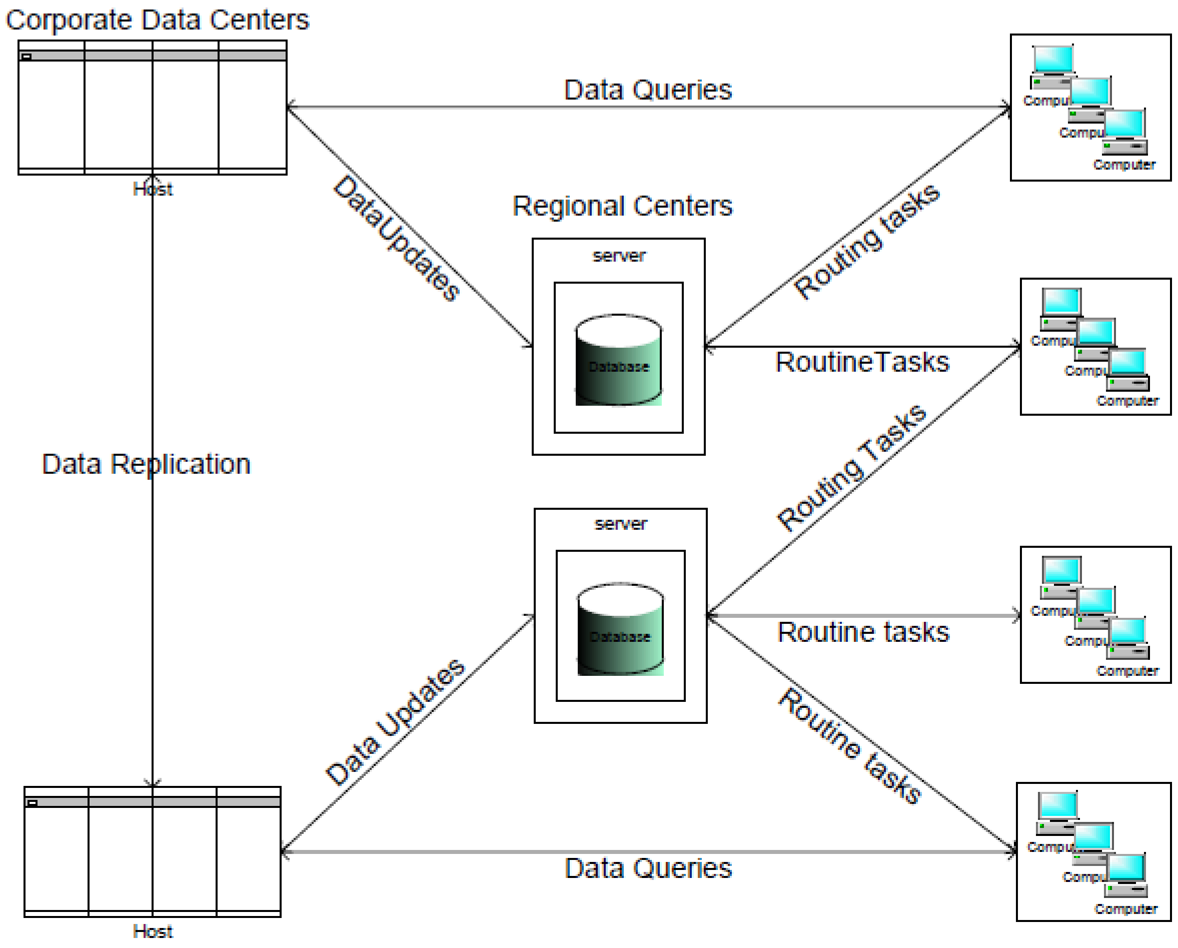


Figure: Distributed Data Centers

Client/Server Model

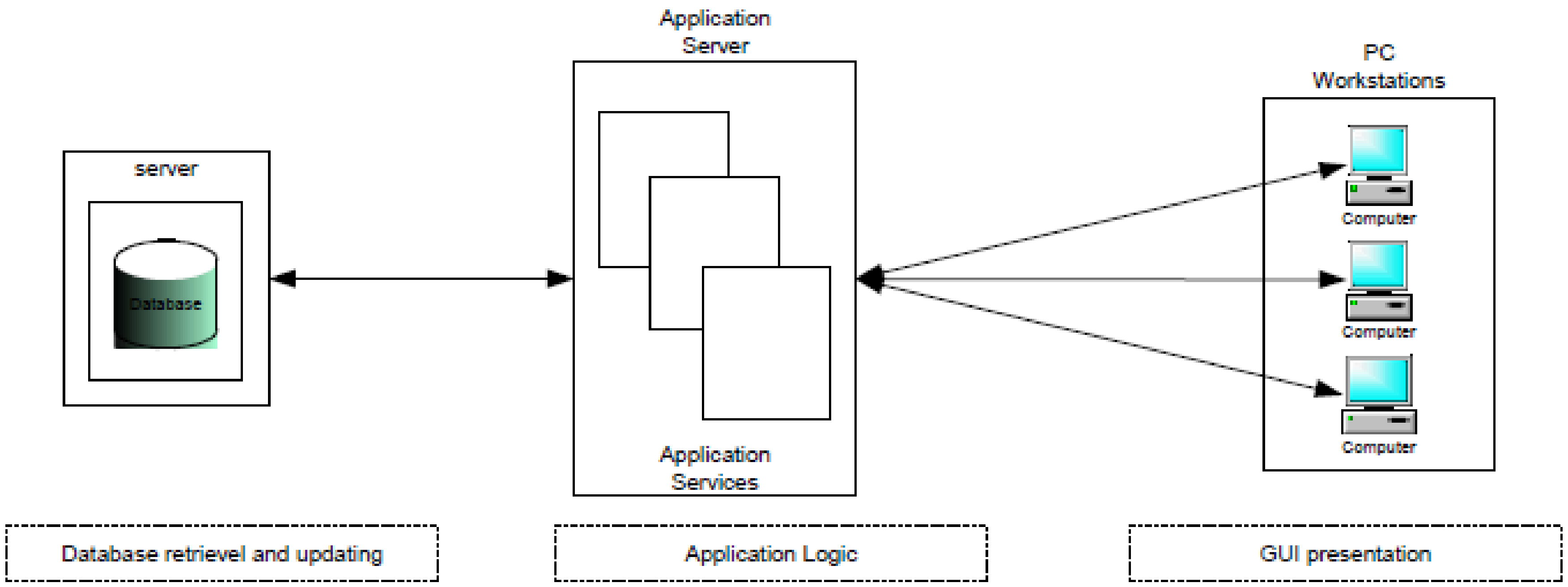


Figure: Client/Server 3-Tier Model

- Complements distributed systems
- Responds to limitations found in the two host data processing models:
  1. The traditional mainframe host model, in which a single mainframe provides shared data access to many dumb terminals,
  2. The local area network (LAN) model, in which many isolated systems access a file server that provides no central processing power.
- Provides integration of data and services
- Application Processing provided by multiple tiers
  1. Database Server
  2. Application Server
  3. PC Workstation

Distributed Client/Server Model

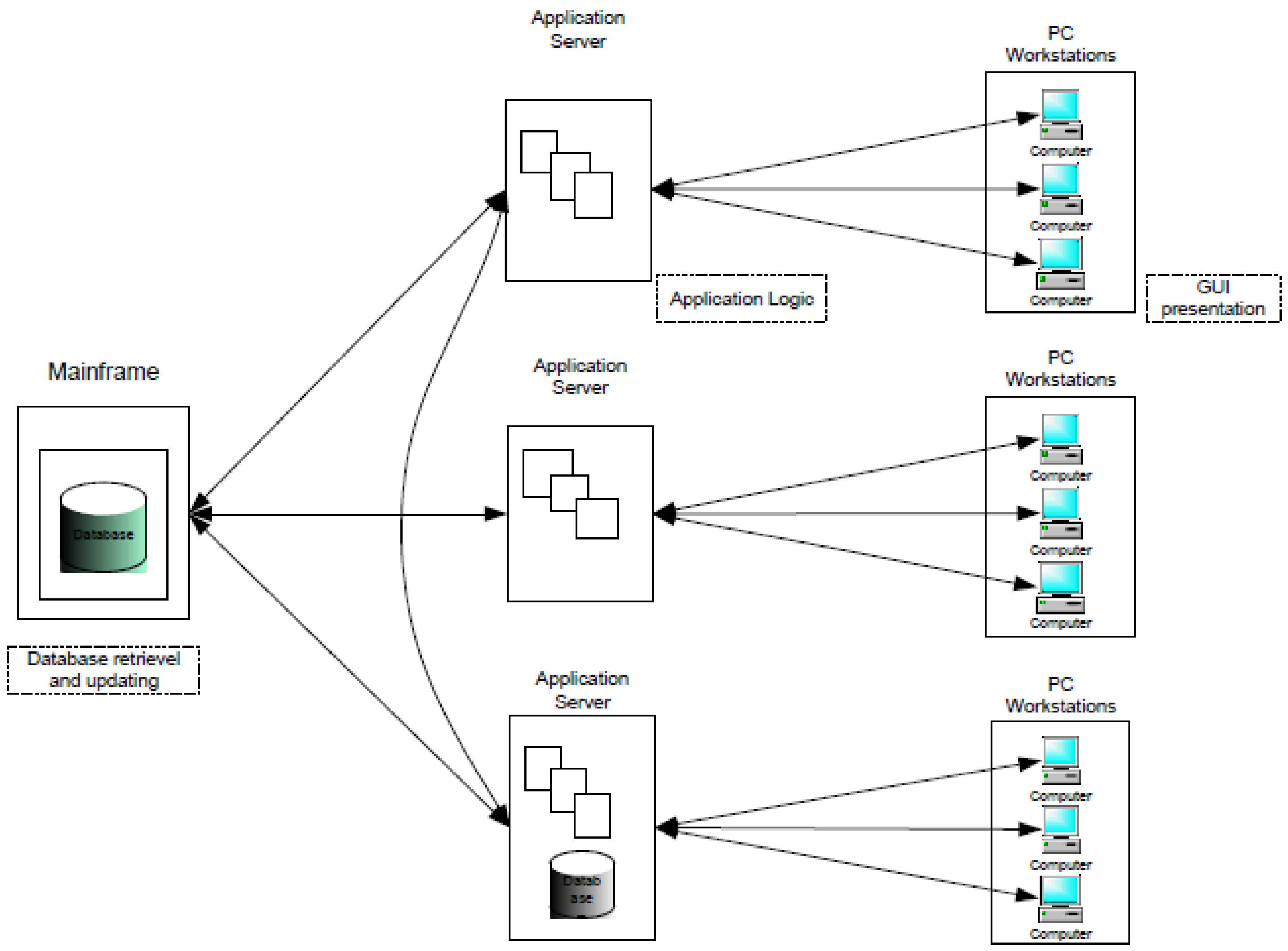


Figure: Distributed Client/Server Model

- Application processing provided by all tiers of the network
  1. Mainframe
  2. Application Servers
  3. Workstations
- Multiple databases to support distributed data requirements
- Supports high volume, load balancing and scalability (extendibility)
- Requires extensive network administration and application management.

Inter-process Communication

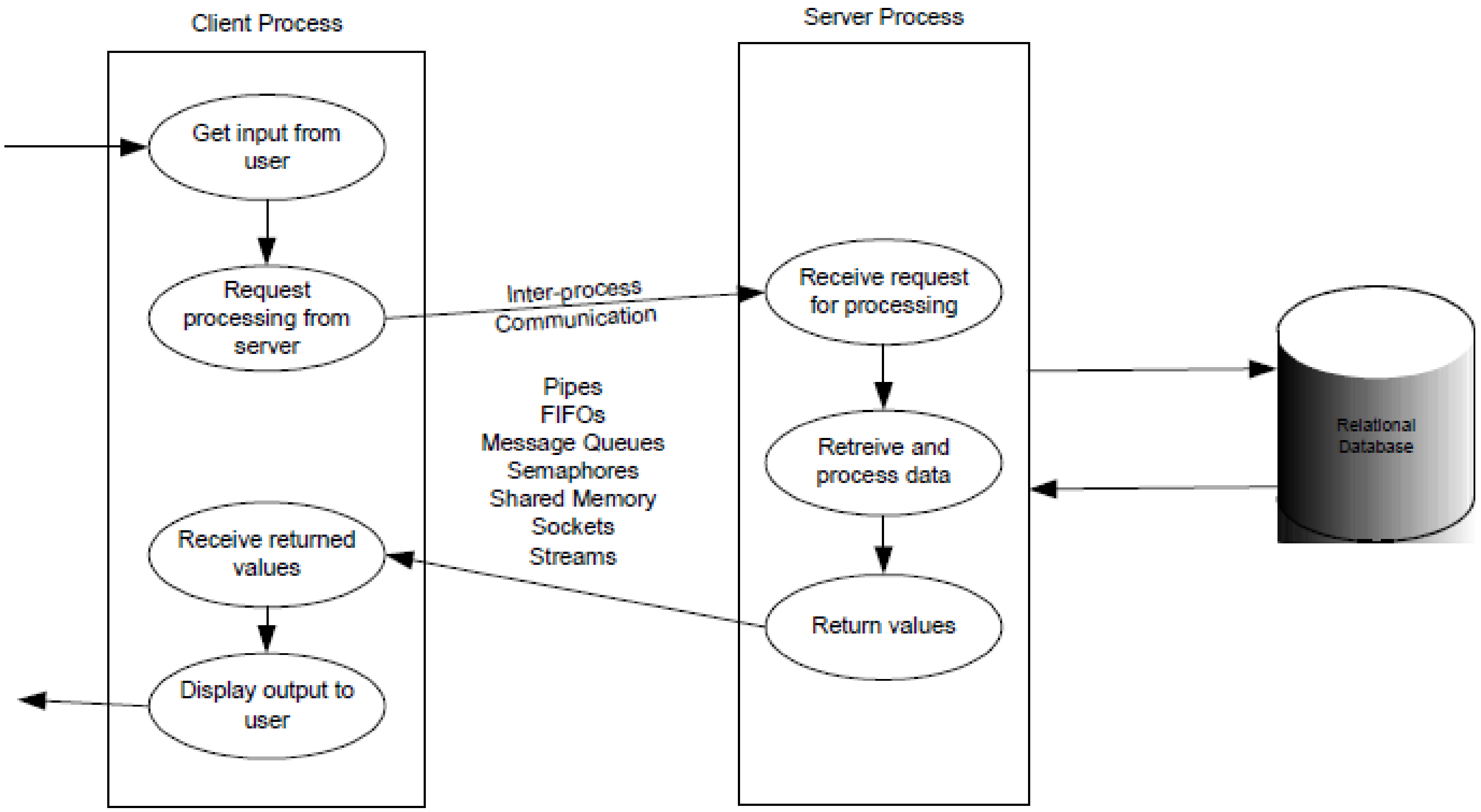


Figure: Inter-Process Communication

- Basis for client/server computing
- Client process communicates with server process
- Each process performs separate functions
- Data is passed between processes using IPC functions

Benefits of the Client/Server Model

- Divides Application Processing across multiple machines:
    - Non-critical data and functions are processed on the client
    - Critical functions are processed on the server
- Optimizes Client Workstations for data input and presentation (e.g., graphics and mouse support)
- Optimizes the Server for data processing and storage (e.g., large amount of memory and disk space)

- Scales Horizontally: Multiple servers, each server having capabilities and processing power, can be added to distribute processing load.
- Scales Vertically: Can be moved to more powerful machines, such as minicomputer or a mainframe to take advantage of the larger system’s performance
- Reduces Data Replication: Data stored on the servers instead of each client, reducing the amount of data replication for the application.

Client/Server 2-Tier Architecture

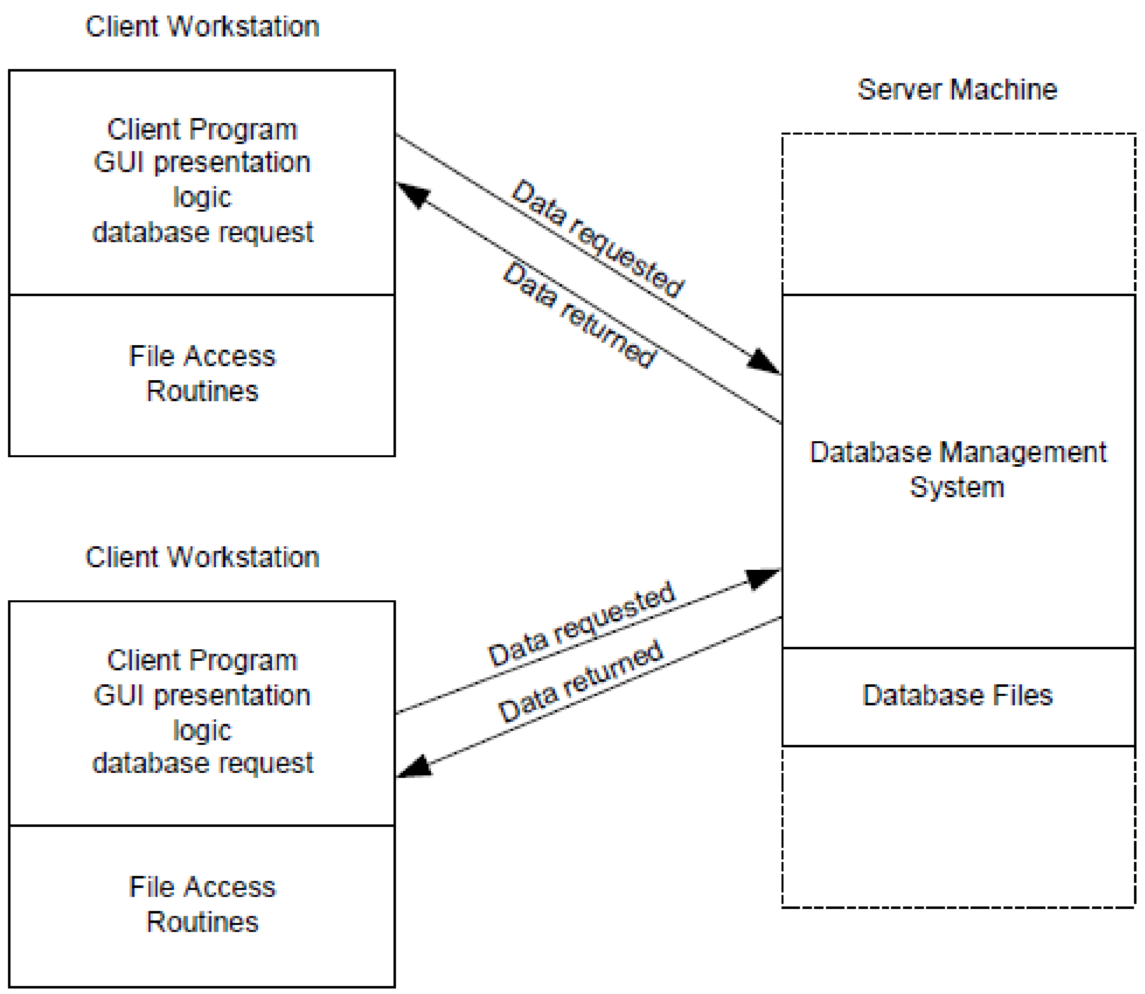
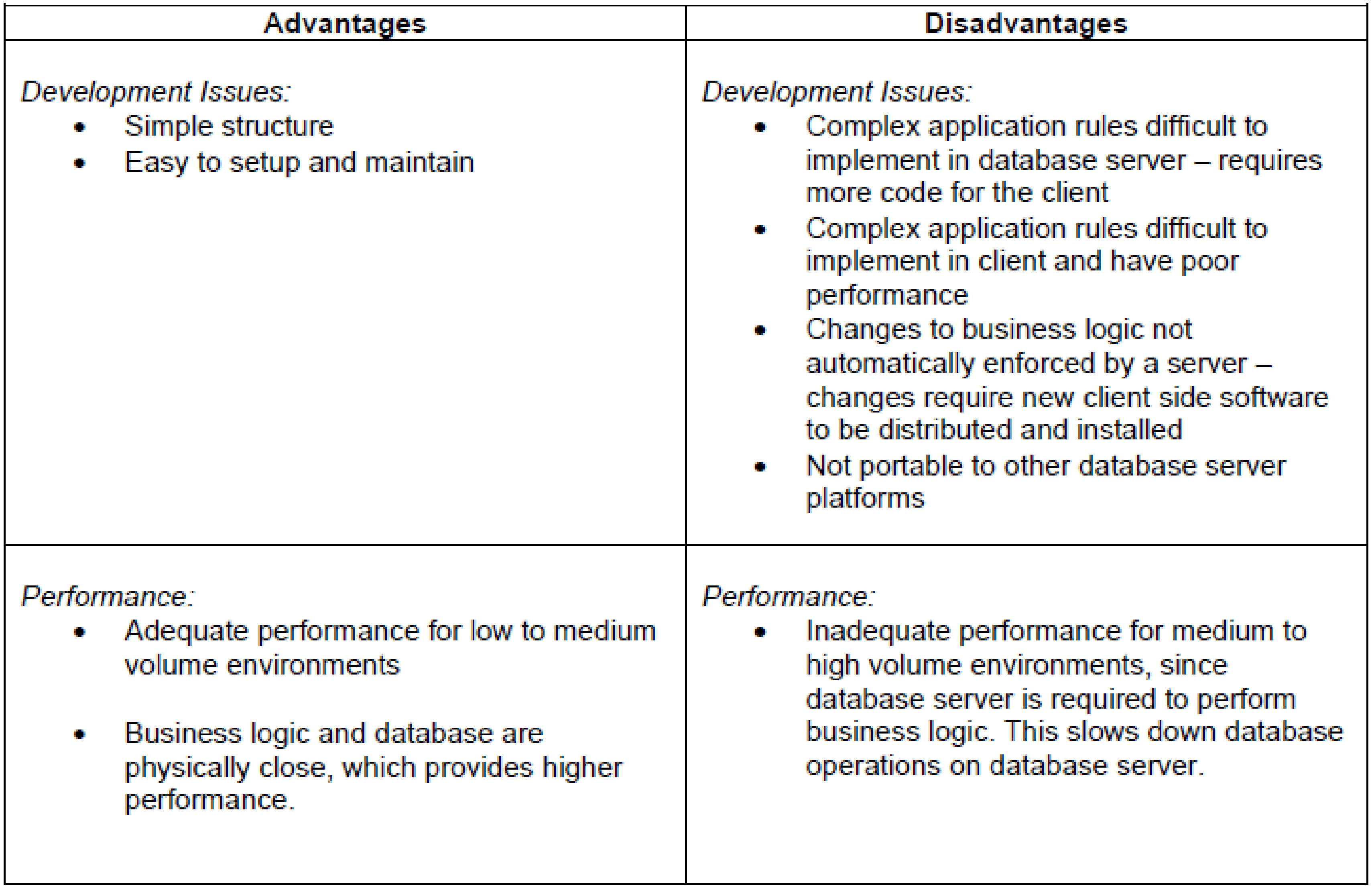


Figure: Client/Server 2-Tier Architecture

Two – Tier Pros and Cons



Advantages	Disadvantages
Development Issues: • Simple structure • Easy to setup and maintain	*Development Issues:* • Complex application rules difficult to implement in database server – requires more code for the client • Complex application rules difficult to implement in client and have poor performance • Changes to business logic not automatically enforced by a server – changes require new client side software to be distributed and installed • Not portable to other database server platforms
Performance: • Adequate performance for low to medium volume environments • Business logic and database are physically close, which provides higher performance.	*Performance:* • Inadequate performance for medium to high volume environments, since database server is required to perform business logic. This slows down database operations on database server.

3-Tier Client/Server Architecture

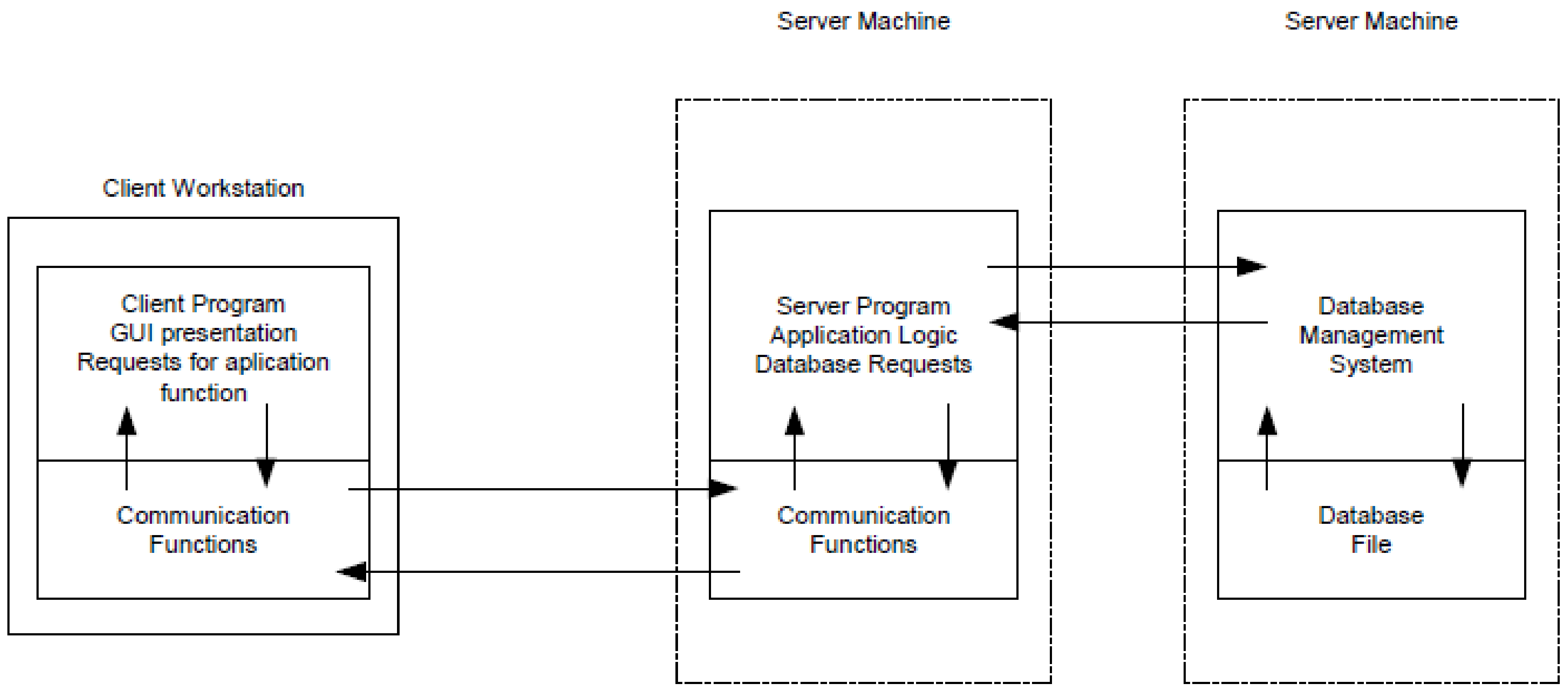
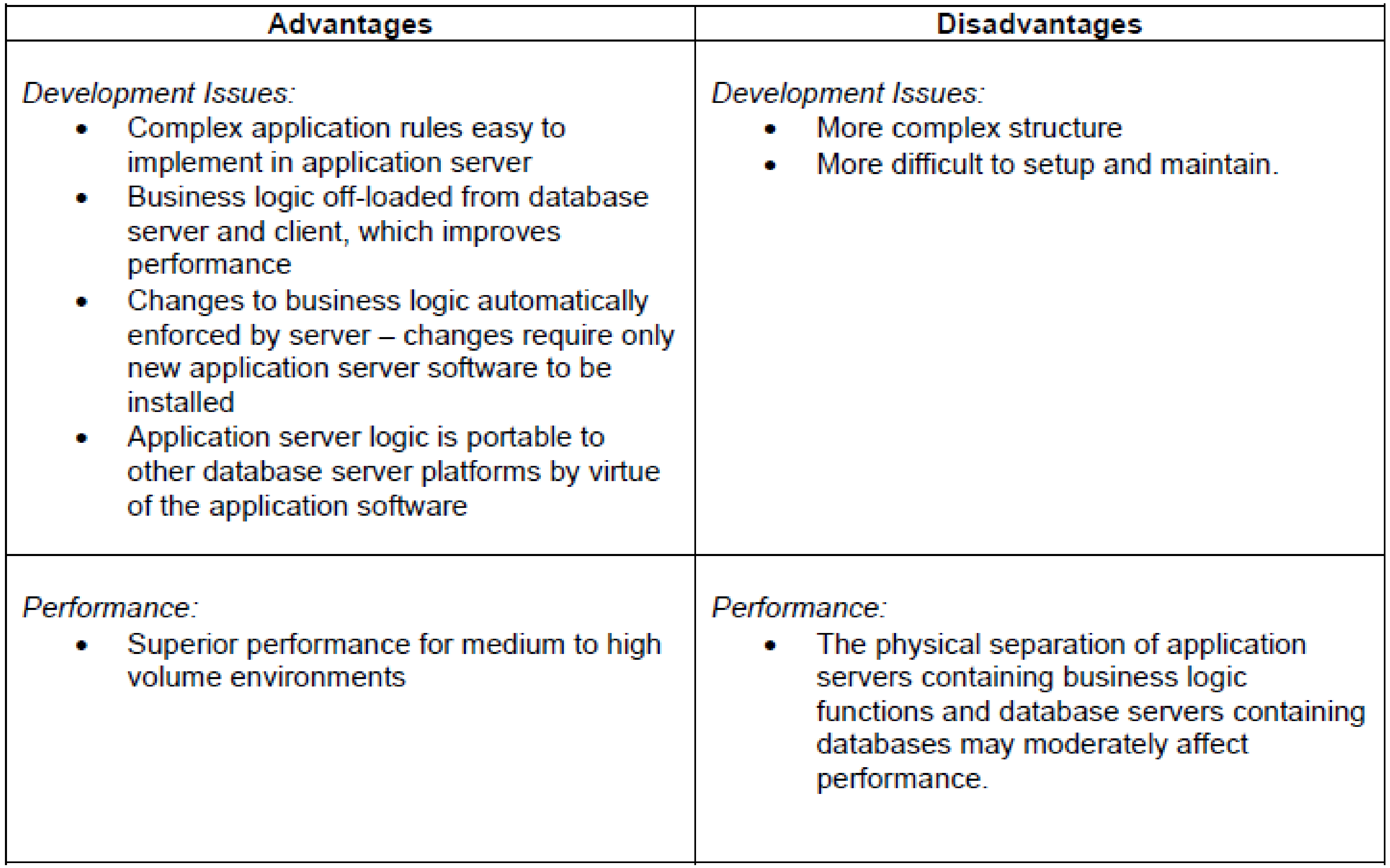


Figure : Typical 3 – Tier Architecture

- 3-Tier client-server architectures have 3 essential components:
  1. A Client PC
  2. An Application Server
  3. A Database Server
- 3-Tier Architecture Considerations:
- Client program contains presentation logic only
  - Less resources needed for client workstation
  - No client modification if database location changes
  - Less code to distribute to client workstations
- One server handles many client requests
  - More resources available for server program
  - Reduces data traffic on the network

3 – Tier Pros and Cons



Advantages	Disadvantages
Development Issues: • Complex application rules easy to implement in application server • Business logic off-loaded from database server and client, which improves performance • Changes to business logic automatically enforced by server – changes require only new application server software to be installed • Application server logic is portable to other database server platforms by virtue of the application software	*Development Issues:* • More complex structure • More difficult to setup and maintain.
Performance: • Superior performance for medium to high volume environments	*Performance:* • The physical separation of application servers containing business logic functions and database servers containing databases may moderately affect performance.

Middleware

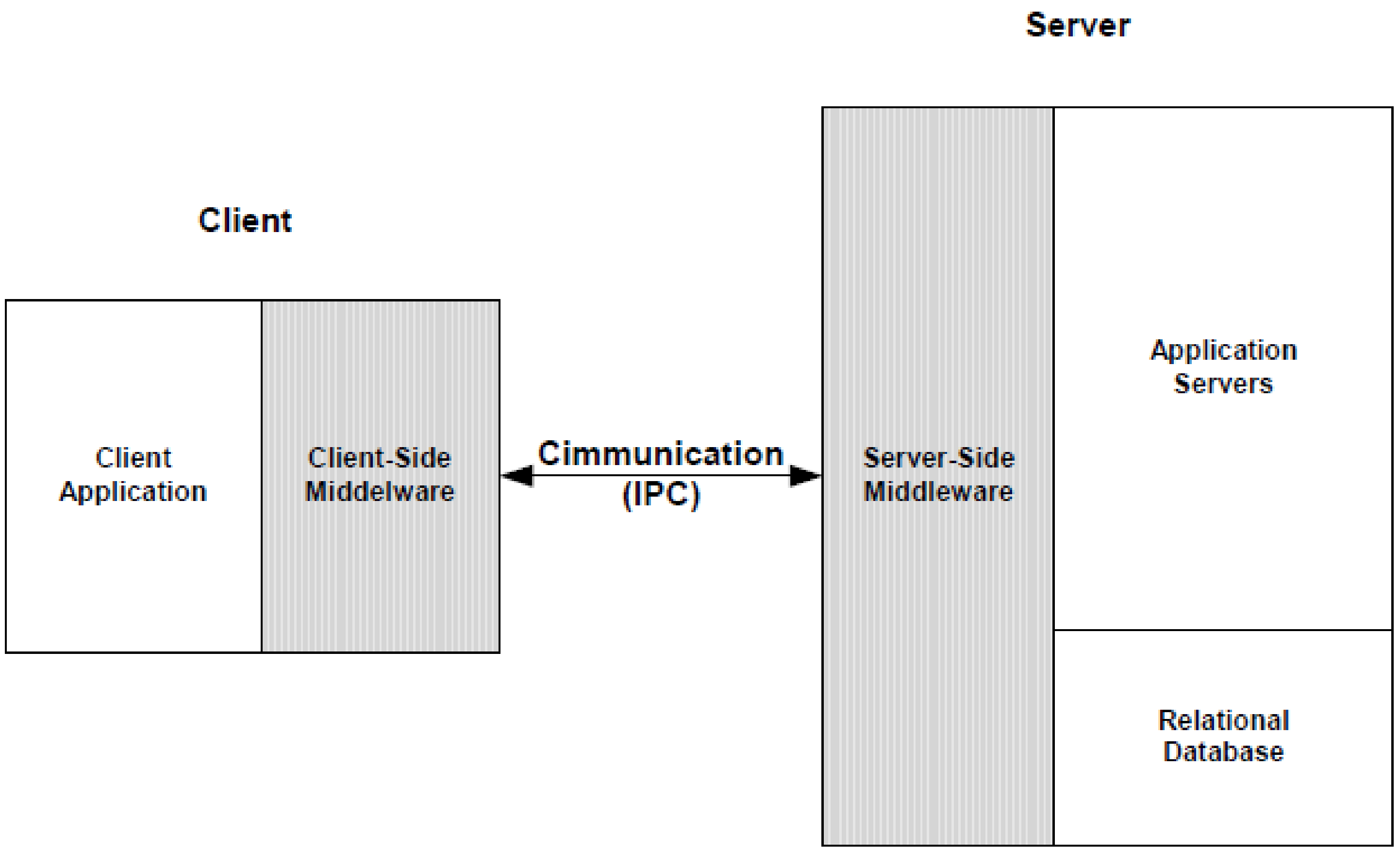


Figure: Middleware

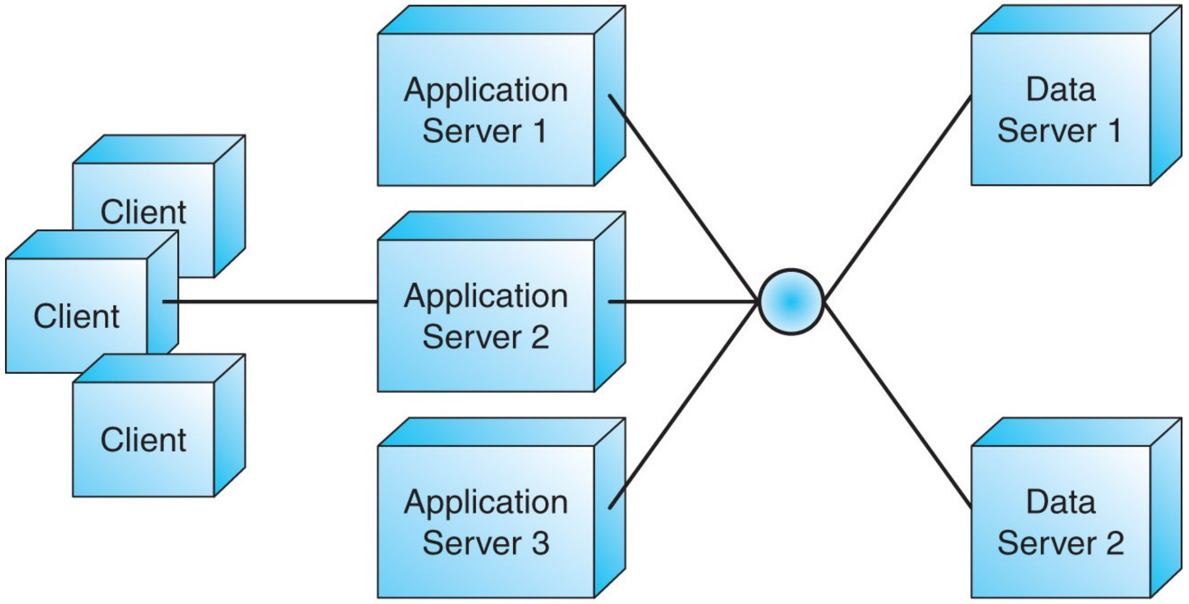
- Simplifies 3-tier application development and administration by providing an extra application server layer to manage communication between components.
- **Middleware Characteristics:**
  - Simplifies partitioning of application processing among clients and servers
  - Manages distributed transactions among multiple databases
  - Communicates with heterogeneous database products within a single application.
  - Supports application scalability
  - Supports service requests prioritization, load-balancing, data dependant routing and queuing.

Multitier (N-tier) C-S Architecture

- A multitier (N-tier) architecture is an 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

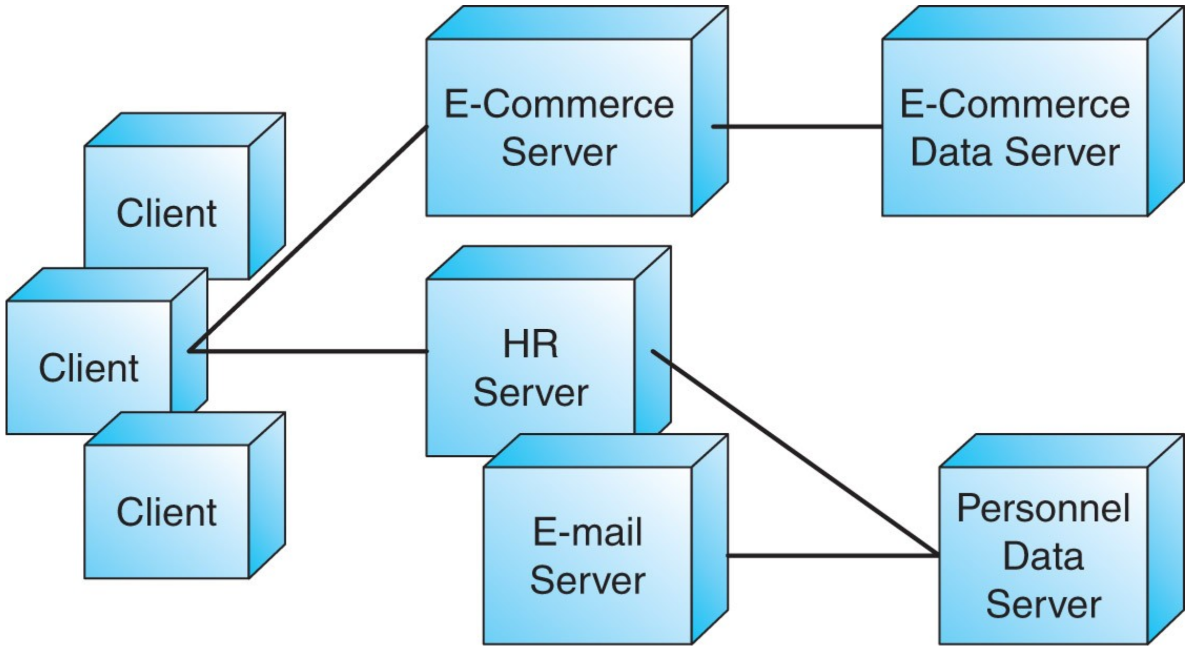
Replication

- Application and data servers are replicated
- Servers share the total workload



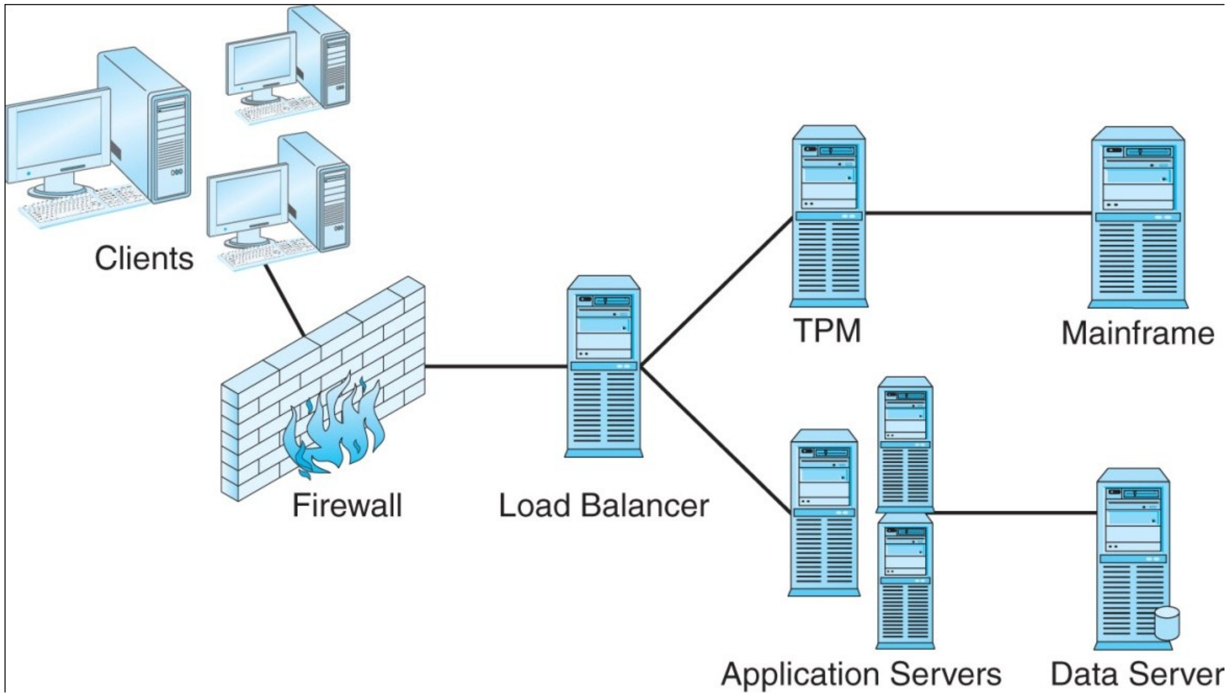
Specialization

- Servers are specialized
- Each server handles a designated part of the workload, by function



Portal Services

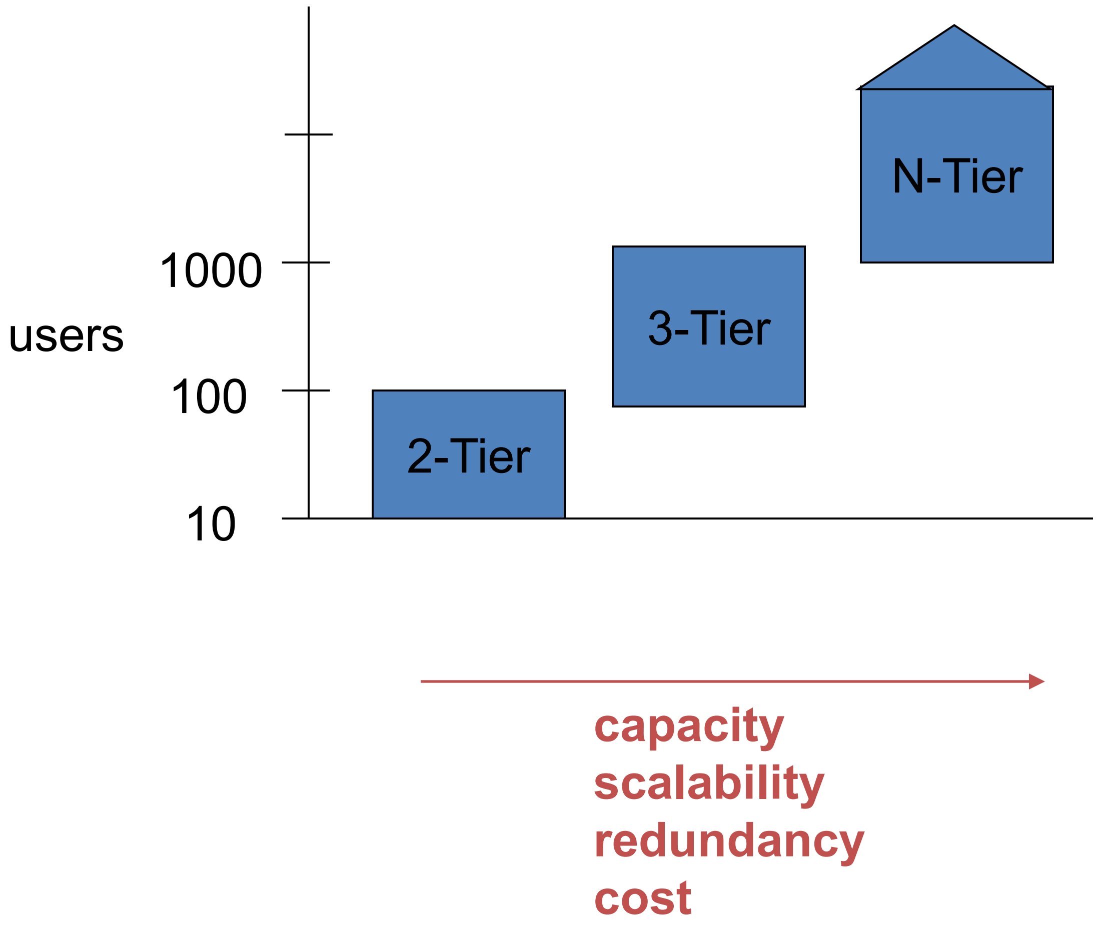
- Portal servers handle incoming traffic, reducing application server load
  - e.g., firewall, load balancer, transaction processing manager



N-Tier Characteristics

- Advantages
  - Decoupling of software components
  - Flexibility to add/remove platforms in response to load
  - Scalability
  - Easy to manage
  - Better and finer security control to the whole system
- Disadvantages
  - Higher costs (maintenance, design, electrical load, cooling)
  - Response time of the system may be slower
- Typical Application
  - 1000+ users
  - Large business or organization

Characteristics Summary



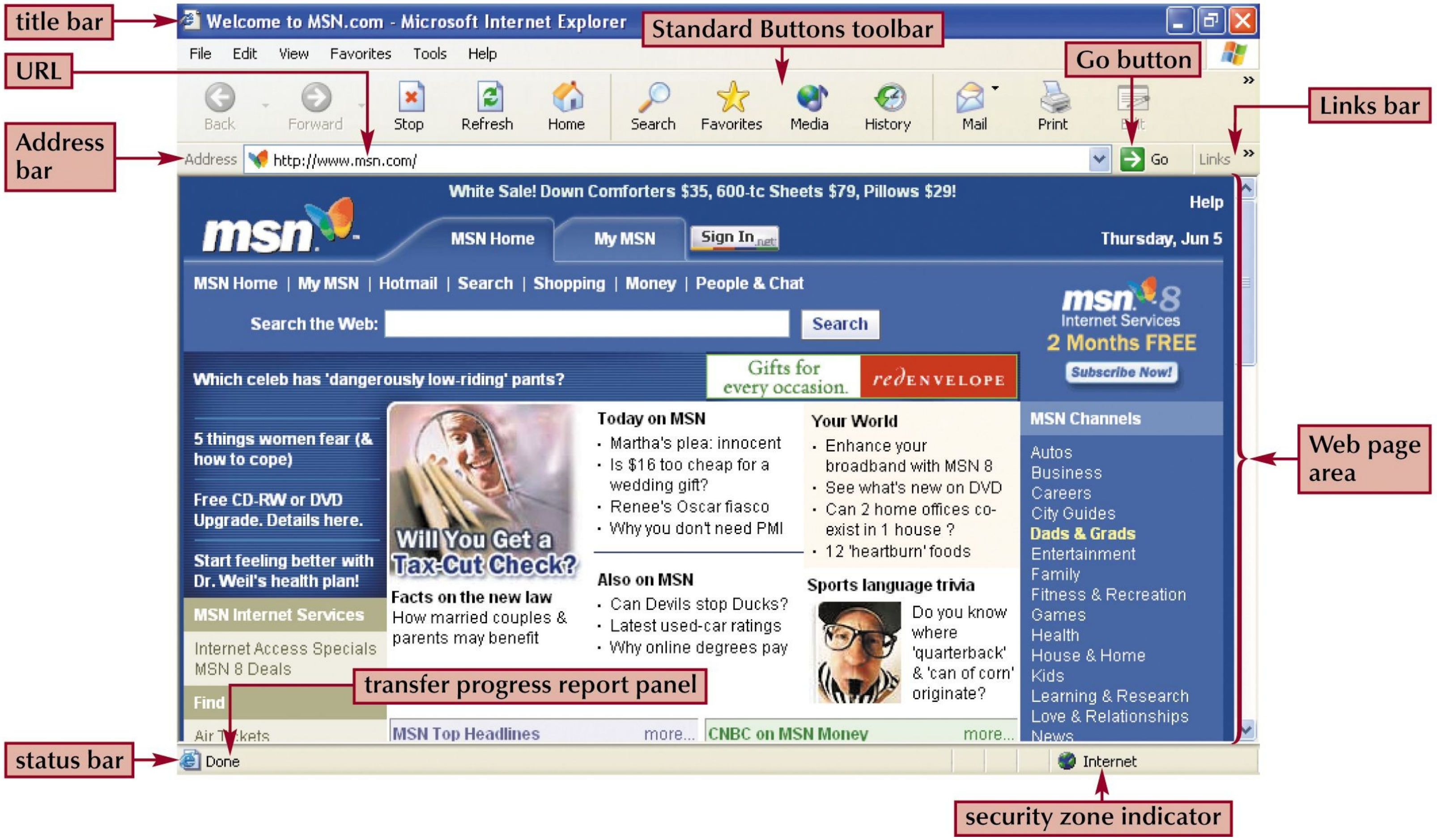
- large e-commerce, business, or organization
- small e-commerce, regional business or organization
- local business or organization

Universal Internet Browsing

- A web browser or Internet browser is a software application for retrieving, presenting, and traversing information resources on the World Wide Web.
- An *information resource* is identified by a Uniform Resource Identifier (URI) and may be a web page, image, video, or other piece of content.
- Although browsers are primarily intended to access the World Wide Web, they can also be used to access information provided by Web servers in private networks or files in file systems.
- Hyperlinks present in resources enable users to easily navigate their browsers to related resources.

The Microsoft Internet Explorer window

MSN home page displayed in Internet Explorer **Figure 4**


title bar
URL
Address bar
status bar
Standard Buttons toolbar
Go button
Links bar
Web page area
transfer progress report panel
security zone indicator
Welcome to MSN.com - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Forward Stop Refresh Home Search Favorites Media History Mail Print
Address http://www.msn.com/
Go Links
Done
Internet

White Sale! Down Comforters $35, 600-tc Sheets $79, Pillows $29!

Help

MSN Home | My MSN | Sign In

Thursday, Jun 5

MSN Home | My MSN | Hotmail | Search | Shopping | Money | People & Chat

Search the Web: Search

msn 8 Internet Services 2 Months FREE Subscribe Now!

Which celeb has 'dangerously low-riding' pants?

Gifts for every occasion. RedEnvelope

5 things women fear (& how to cope)

Free CD-RW or DVD Upgrade. Details here.

Start feeling better with Dr. Weil's health plan!

MSN Internet Services

Internet Access Specials
MSN 8 Deals

Find

Air Tickets

Will You Get a Tax-Cut Check?

Facts on the new law
How married couples & parents may benefit

Today on MSN
- Martha's plea: innocent
- Is $16 too cheap for a wedding gift?
- Renee's Oscar fiasco
- Why you don't need PMI

Also on MSN
- Can Devils stop Ducks?
- Latest used-car ratings
- Why online degrees pay

Your World
- Enhance your broadband with MSN 8
- See what's new on DVD
- Can 2 home offices co-exist in 1 house ?
- 12 'heartburn' foods

Sports language trivia
Do you know where 'quarterback' & 'can of corn' originate?

MSN Top Headlines more...
CNBC on MSN Money more...

MSN Channels
Autos
Business
Careers
City Guides
Dads & Grads
Entertainment
Family
Fitness & Recreation
Games
Health
House & Home
Kids
Learning & Research
Love & Relationships
News

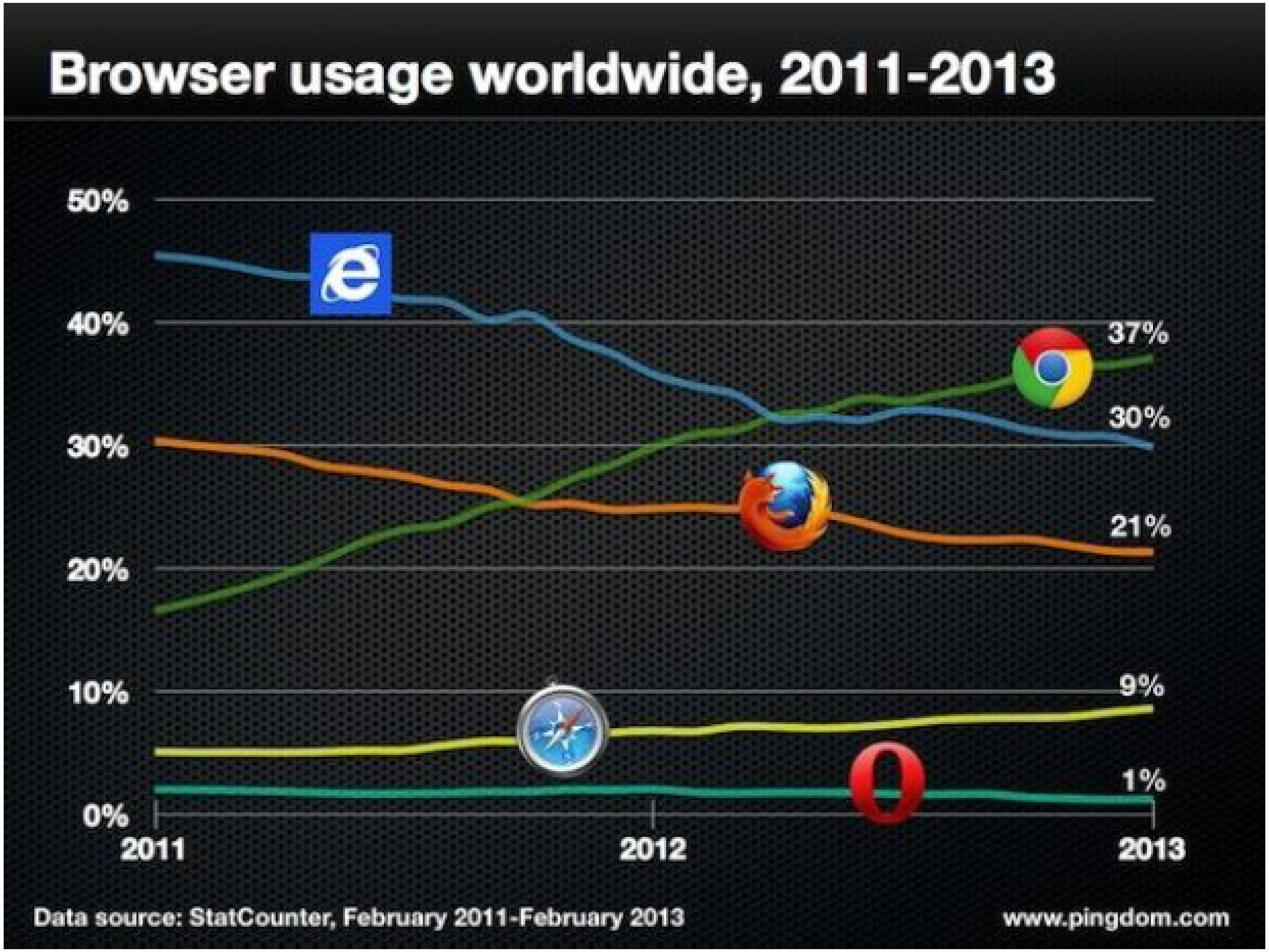
History and Development

- Tim Berners-Lee developed it in December of 1990. It was released in March, 1991.
- The first commonly available web browser with a graphical user interface was Erwise.
- The development of Erwise was initiated by Robert Cailliau.
- In 1993, browser software was further innovated by Marc Andreessen with the release of Mosaic, "the world's first popular browser”.
- Microsoft responded with its Internet Explorer in 1995.
  - Internet Explorer gained dominance in the web browser market; Internet Explorer usage share peaked at over 95% by 2002

- Opera debuted in 1996
- In 1998, Netscape launched what was to become the Mozilla Foundation in an attempt to produce a competitive browser using the open source software model. That browser eventually evolve into Firefox
- Apple's Safari had its first beta release in January 2003.
- The most recent major entrant to the browser market is Chrome, first released in September 2008.
  - Chrome's take-up has increased significantly year by year, by doubling its usage
- Internet Explorer will be deprecated in Windows 10, with Microsoft Edge replacing it as the default web browser

Figure: Most widely used browsers

Some Statistics

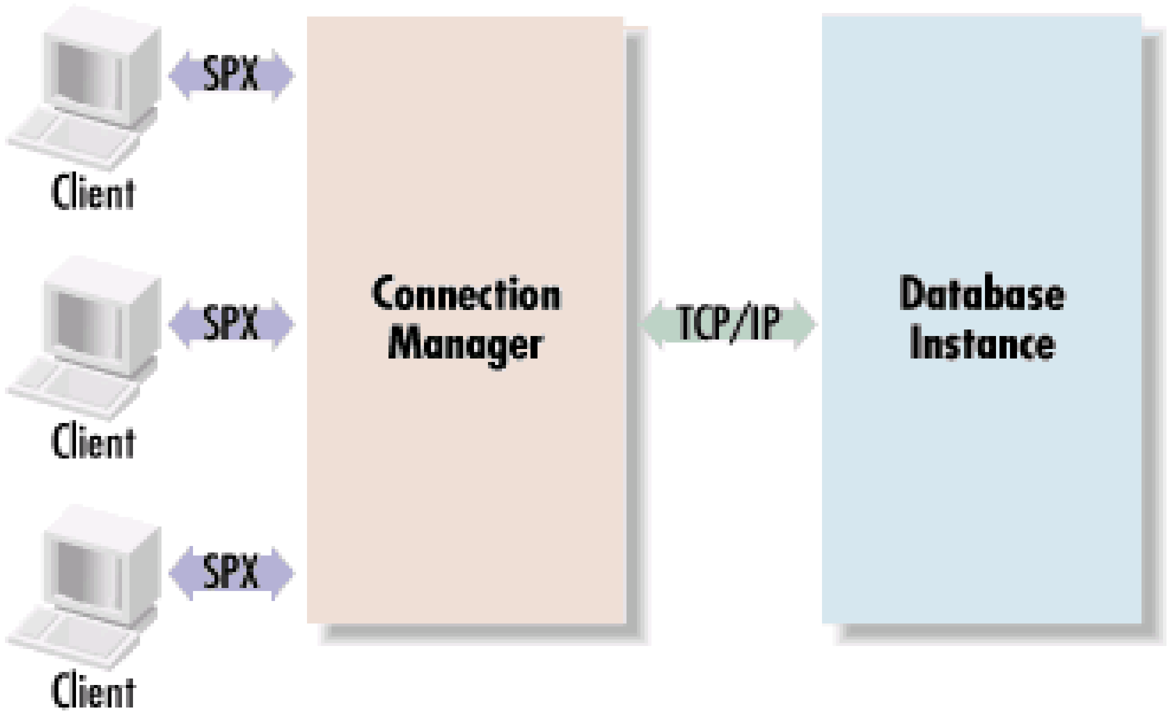


Client/Server Structure of the Web

- Web is a collection of files that reside on computers, called **Web servers**, that are located all over the world and are connected to each other through the Internet.
- When you use your Internet connection to become part of the Web, your computer becomes a **Web client** in a worldwide client/server network.
- A **Web browser** is the software that you run on your computer to make it work as a web client.

Multiprotocol Support

- Multi-protocol support allows clients using one network protocol to connect to a database server or any other client/server using a different network protocol.
- Example



MPLS

- Multi-Protocol Label Switching
- Layer 2.5 networking protocol
- In the traditional OSI model:
  - Layer 2 covers protocols like Ethernet, which can carry IP packets, but only over simple LANs or point-to-point WANs.
  - Layer 3 covers Internet-wide addressing and routing using IP protocols.
  - MPLS sits between these traditional layers, providing additional features for the transport of data across the network.

- In a traditional IP network:
  - Each router performs an IP lookup (“routing”), determines a next-hop
  - based on its routing table, and forwards the packet to that next-hop.
  - Rinse and repeat for every router, until the final destination is reached.
- MPLS does “label switching” instead:
  - The first device does a routing lookup, just like before:
    - ➢But instead of finding a next-hop, it finds the final destination router.
    - ➢And it finds a pre-determined path from “here” to that final router.

- The router applies a “label” based on this information.
- Future routers use the label to route the traffic
  - Without any additional IP lookups.
- At the final destination router the label is removed.
  - the packet is delivered via normal IP routing

Thank You

If you have any Queries write to me

@

Jalauddin.mansur@gmail.com

jalawdarling@hotmail.com