and disconnected from a network (like ...). DHCP provides temporary IP addresses for a limited period of time.

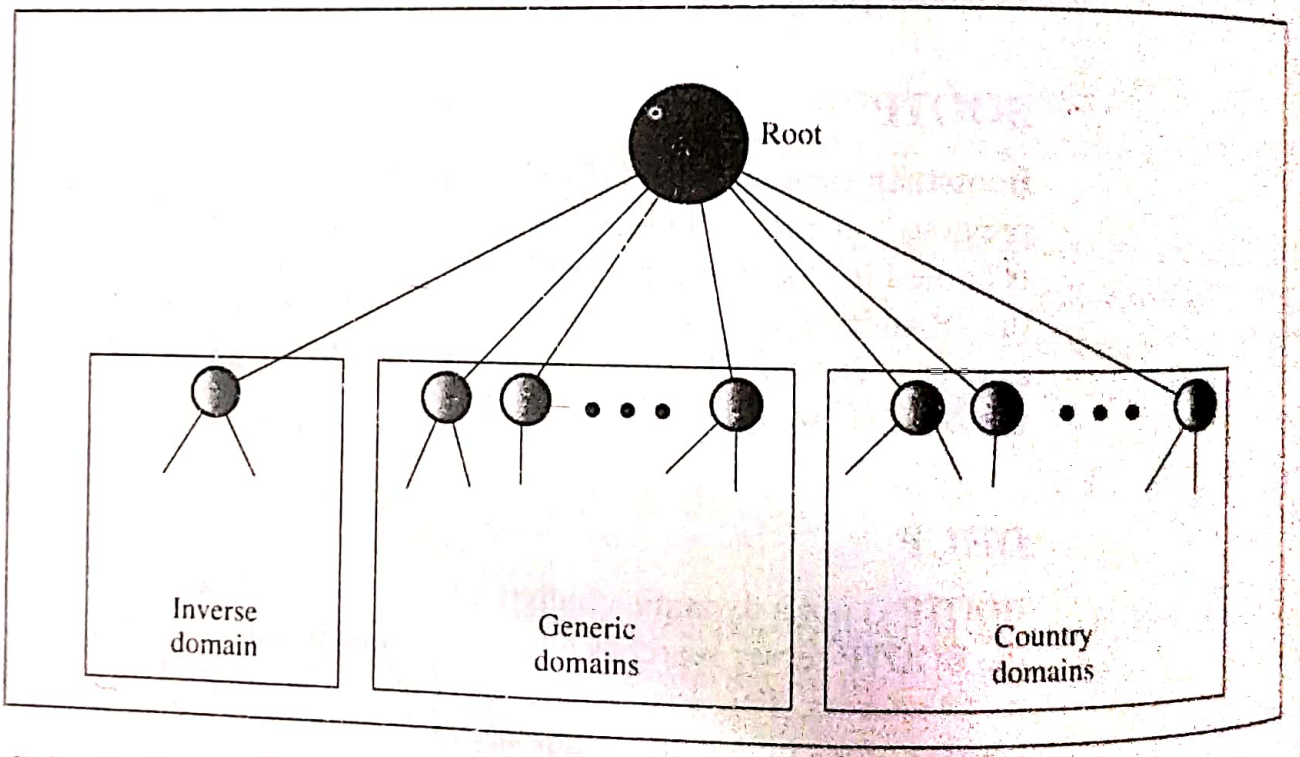
25.3 DOMAIN NAME SYSTEM (DNS)

To identify an entity, TCP/IP protocols use the IP address, which uniquely identifies the connection of a host to the Internet. However, people prefer to use names instead of addresses. Therefore, we need a system that can map a name to an address and conversely an address to a name. In TCP/IP, this is the **Domain Name System (DNS)**.

DNS in the Internet

DNS is a protocol that can be used in different platforms. In the Internet, the domain name space (tree) is divided into three different sections: generic domains, country domains, and inverse domain (see Figure 25.3).

Figure 25.3 *DNS in the Internet*



Generic Domains

The **generic domains** define registered hosts according to their generic behavior. Each node in the tree defines a domain, which is an index to the domain name space database (see Figure 25.4).

Looking at the tree, we see that the first level in the generic domain section allows seven possible three-character labels. These labels describe the organization types as listed in Table 25.1.

Recently a few more first-level labels have been proposed; these are shown in Table 25.2.

Figure 25.4

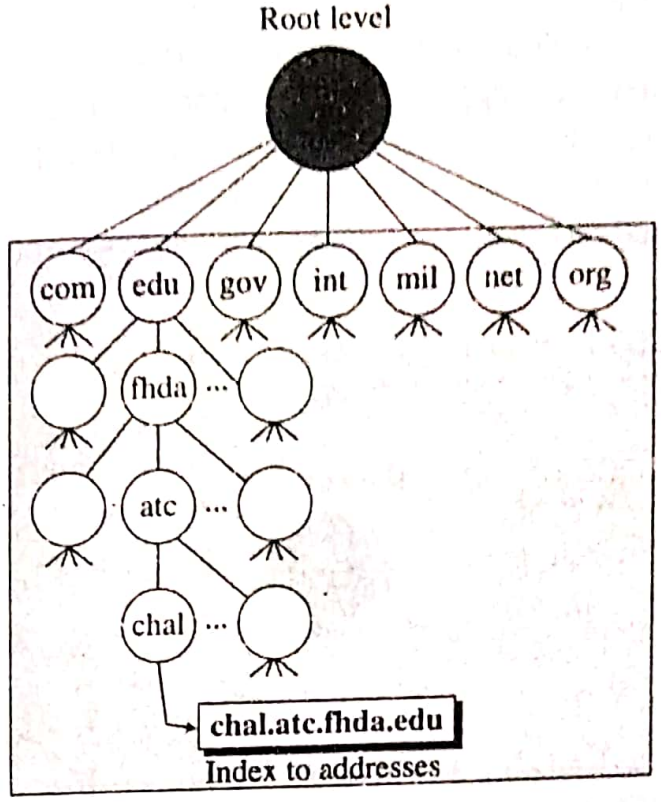


Table 25.1 *Generic domain labels*

Label	*Description*
com	Commercial organizations
edu	Educational institutions
gov	Government institutions
int	International organizations
mil	Military groups
net	Network support centers
org	Nonprofit organizations

Table 25.2 *Proposed generic domain labels*

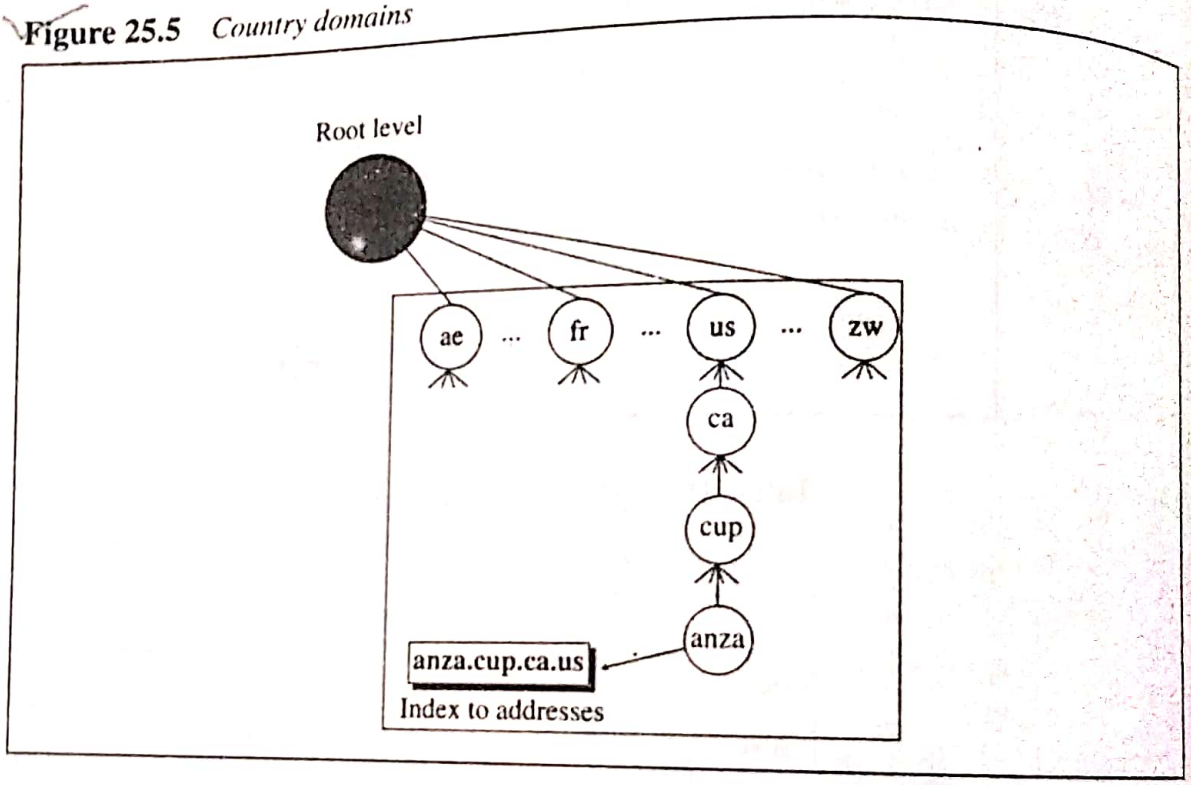
Label	*Description*
arts	Cultural organizations
firm	Businesses or firms
info	Information service providers
nom	Personal nomenclatures
rec	Recreation/entertainment organizations
store	Businesses offering goods to purchase
web	Web-related organizations

Country Domains

The **country domain** section follows the same format as the generic domains but uses two-character country abbreviations (e.g., "us" for United States) in place of the three-character organizational abbreviations at the first level. Second-level labels can be organizational, or they can be more specific, national designations. The United States, for example, uses state abbreviations as a subdivision of "us" (e.g., ca.us.).

Figure 25.5 shows the country domain section. The address *anza.cup.ca.us* can be translated to De Anza College in Cupertino in California in the United States.

Figure 25.5 *Country domains*



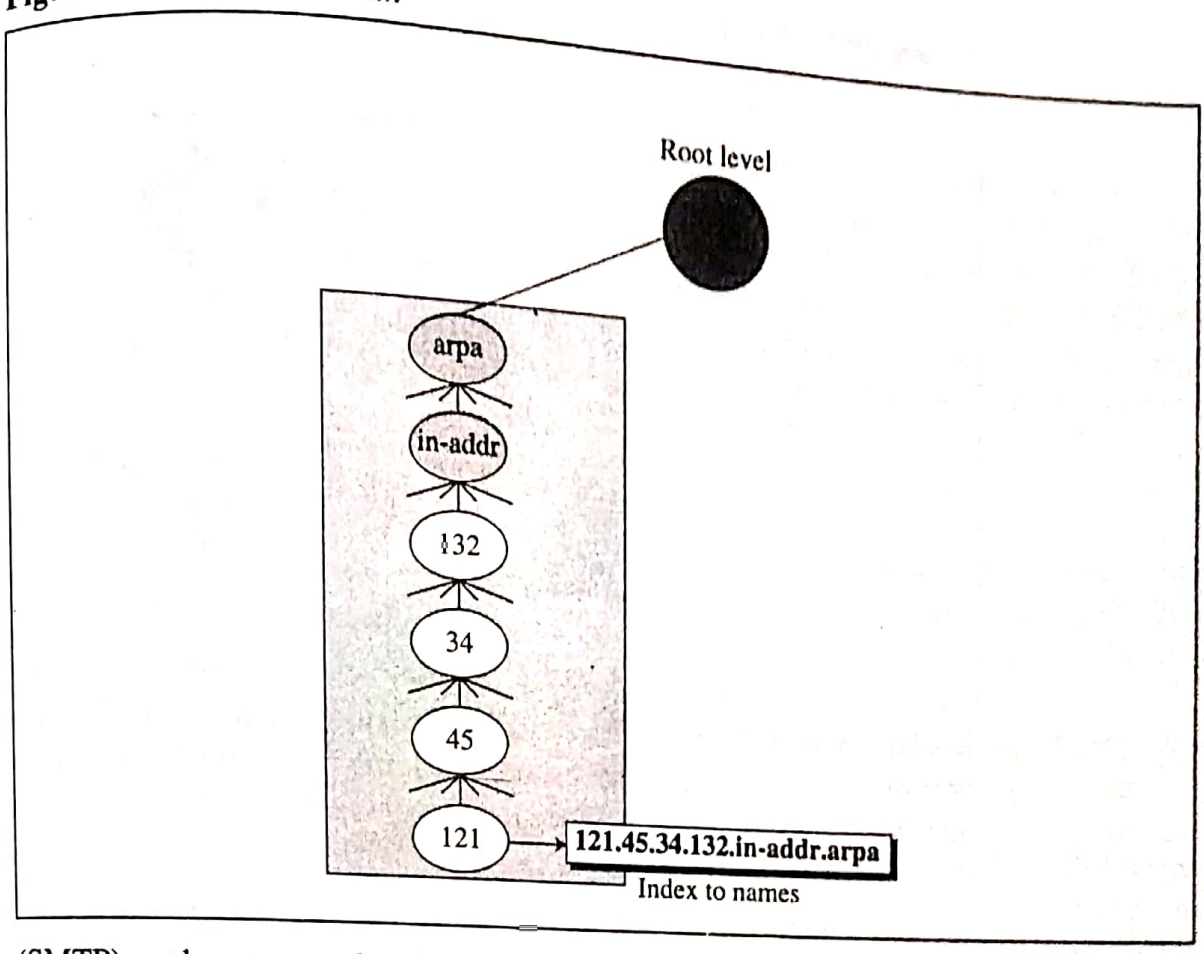
Inverse Domain

The **inverse domain** is used to map an address to a name. This may happen, for example, when a server has received a request from a client to do a task. Whereas the server has a file that contains a list of authorized clients, the server lists only the IP address of the client (extracted from the received IP packet). To determine if the client is on the authorized list, it can send a query to the DNS server and ask for a mapping of address to name. See Figure 25.6.

25.4 TELNET

The main task of the Internet and its TCP/IP protocol suite is to provide services for users. For example, users want to be able to run different application programs at a remote site and create results that can be transferred to their local site. One way to satisfy these demands is to create different client–server application programs for each desired service. Programs such as file transfer programs (FTP and TFTP), e-mail

Figure 25.6 *Inverse domain*



(SMTP), and so on are already available. But it would be impossible to write a specific client–server program for each demand.

The better solution is a general-purpose client–server program that lets a user access any application program on a remote computer; in other words, allow the user to log on to a remote computer. After logging on, a user can use the services available on the remote computer and transfer the results back to the local computer.

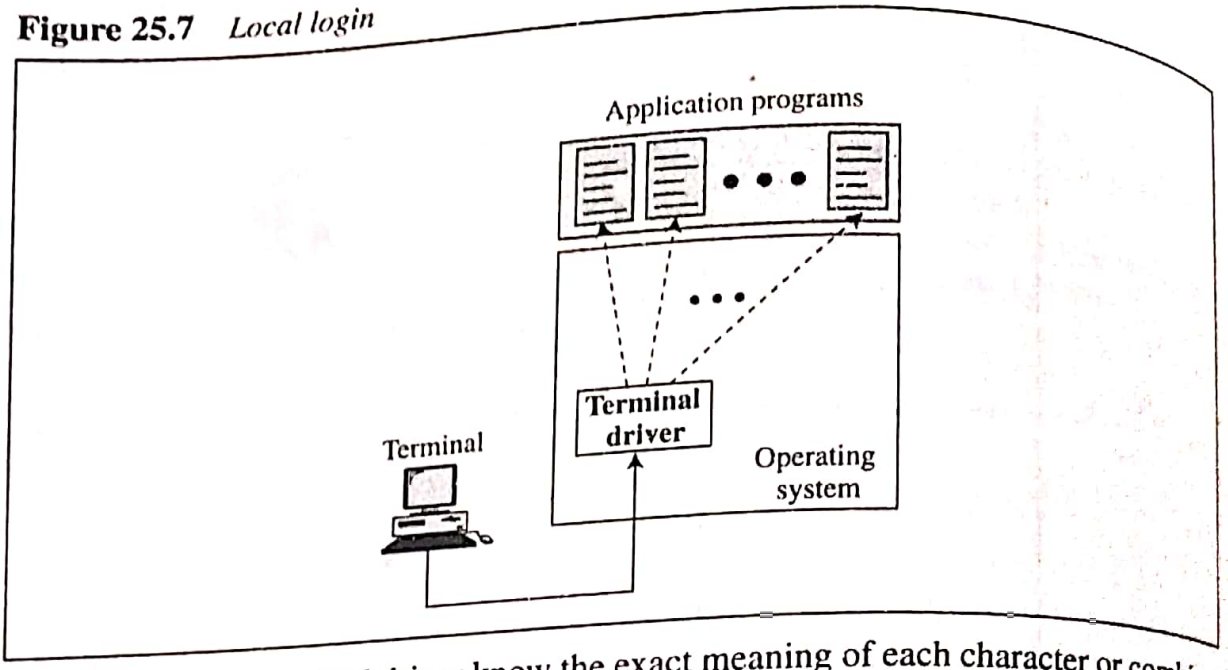
In this section, we discuss a popular client–server application program called TELNET. **TELNET** is an abbreviation for **TErminaL NETwork.** TELNET enables the establishment of a connection to a remote system in such a way that the local terminal appears to be a terminal at the remote system.

TELNET is a general-purpose client–server application program.

Local Login When a user logs into a local time-sharing system, it is called **local login.** As a user types at a terminal or at a workstation running a terminal emulator, the keystrokes are accepted by the terminal driver. The terminal driver passes the characters to the operating system. The operating system, in turn, interprets the combination of characters and invokes the desired application program or utility (see Figure 25.7).

The mechanism, however, is not as simple as it seems because the operating system may assign special meanings to special characters. For example, in UNIX some combinations of characters have special meanings, such as the combination of the control character with the character "z", which means suspend; the combination of the control character with the character "c", which means abort; and so on. Whereas these special situations do not create any problem in local login because the terminal

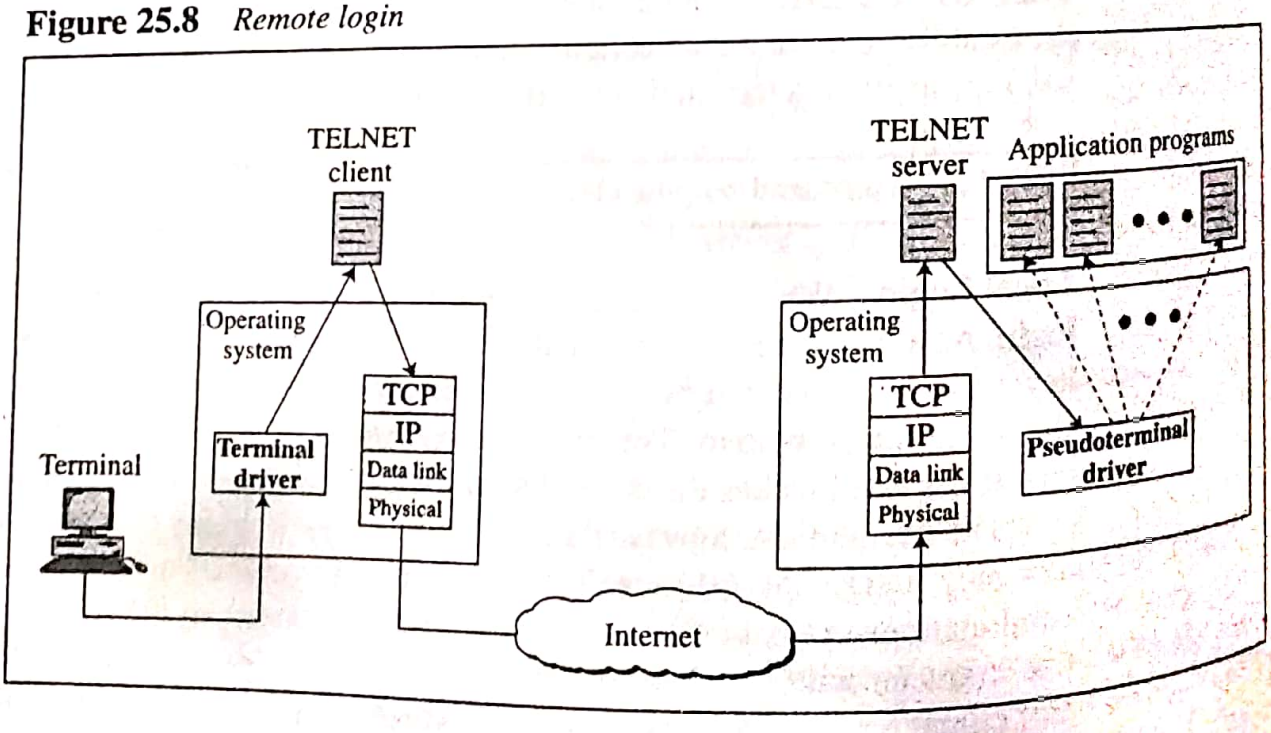
Figure 25.7 *Local login*



emulator and the terminal driver know the exact meaning of each character or combination of characters, they may create problems in remote login. Which process should interpret special characters? The client or the server? We will clarify this situation later in this section.

Remote Login When a user wants to access an application program or utility located on a remote machine, he or she performs **remote login.** Here the TELNET client and server programs come into use. The user sends the keystrokes to the terminal driver where the local operating system accepts the characters but does not interpret them. The characters are sent to the TELNET client, which transforms the characters to a universal character set called *network virtual terminal characters* and delivers them to the local TCP/IP stack (see Figure 25.8).

Figure 25.8 *Remote login*



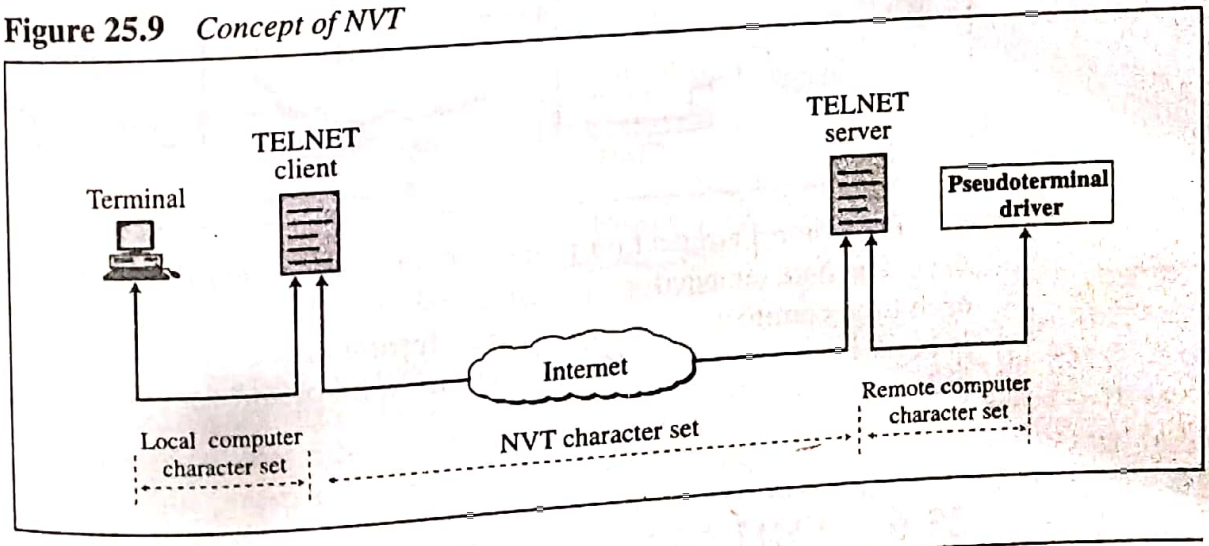
The commands or text, in NVT form, travel through the Internet and arrive at the TCP/IP stack at the remote machine. Here the characters are delivered to the operating system and passed to the TELNET server, which changes the characters to the corresponding characters understandable by the remote computer. However, the characters cannot be passed directly to the operating system because the remote operating system is not designed to receive characters from a TELNET server: it is designed to receive characters from a terminal driver. The solution is to add a piece of software called a *pseudoterminal driver*, which pretends that the characters are coming from a terminal. The operating system then passes the characters to the appropriate application program.

Network Virtual Terminal (NVT)

The mechanism to access a remote computer is complex. This is because every computer and its operating system accepts a special combination of characters as tokens. For example, the end-of-file token in a computer running the DOS operating system is Ctrl+z, while the **UNIX** operating system recognizes Ctrl+d.

We are dealing with heterogeneous systems. If we want to access any remote computer in the world, we first must know to what type of computer we will be connected, and we also must install the specific terminal emulator used by that computer. TELNET solves this problem by defining a universal interface called the **network virtual terminal (NVT)** character set. Via this interface, the client TELNET translates characters (data or commands) that come from the local terminal into NVT form and delivers them to the network. The server TELNET, on the other hand, translates data and commands from NVT form into the form acceptable by the remote computer. For an illustration of this concept, see Figure 25.9.

Figure 25.9 *Concept of NVT*



25.5 FILE TRANSFER PROTOCOL (FTP)

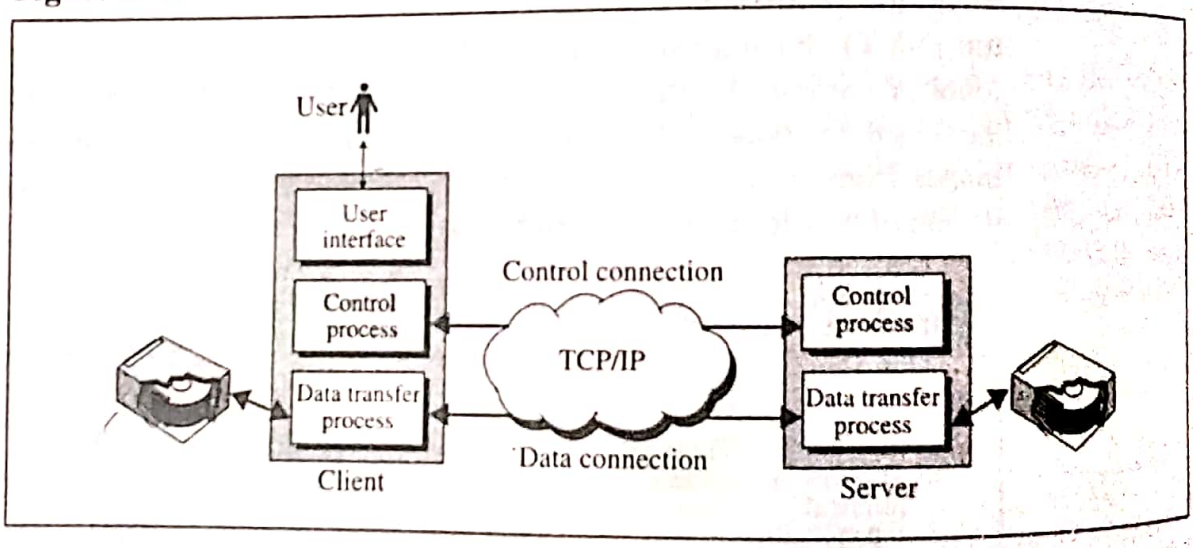
File transfer protocol (FTP) is the standard mechanism provided by TCP/IP for copying a file from one host to another. Transferring files from one computer to another is one of the most common tasks expected from a networking or internetworking environment.

Although transferring files from one system to another seems simple and straightforward, some problems must be dealt with first. For example, two systems may use different file name conventions. Two systems may have different ways to represent text and data. Two systems may have different directory structures. All of these problems have been solved by FTP in a very simple and elegant approach.

FTP differs from other client–server applications in that it establishes two connections between the hosts. One connection is used for **data transfer,** the other for control information (commands and responses). Separation of commands and data transfer makes FTP more efficient. The control connection uses very simple rules of communication. We need to transfer only a line of command or a line of response at a time. The data connection, on the other hand, needs more complex rules due to the variety of data types transferred.

Figure 25.10 shows the basic model of FTP. The client has three components: the user interface, the client control process, and the client data transfer process. The server has two components: the server control process and the server data transfer process. The control connection is made between the control processes. The data connection is made between the data transfer processes.

Figure 25.10 *FTP*



The control connection remains connected during the entire interactive FTP session. The data connection is opened and then closed for each file transferred. It opens each time commands that involve transferring files are used, and it closes when the file is transferred. The two FTP connections, control and data, use different strategies and different port numbers.

25.6 TRIVIAL FILE TRANSFER PROTOCOL (TFTP)

There are occasions when we need to simply copy a file without the need for all of the functionalities of the FTP protocol. For example, when a diskless workstation or a router is booted, we need to download the bootstrap and configuration files. Here we do

not need all of the sophistication provided in FTP. We just need a protocol that quickly copies the files.

Trivial File Transfer Protocol (TFTP) is designed for these types of file transfer. It is so simple that the software package can fit into the read-only memory of a diskless workstation. It can be used at bootstrap time. TFTP can read or write a file for the client. *Reading* means copying a file from the server site to the client site. *Writing* means copying a file from the client site to the server site.

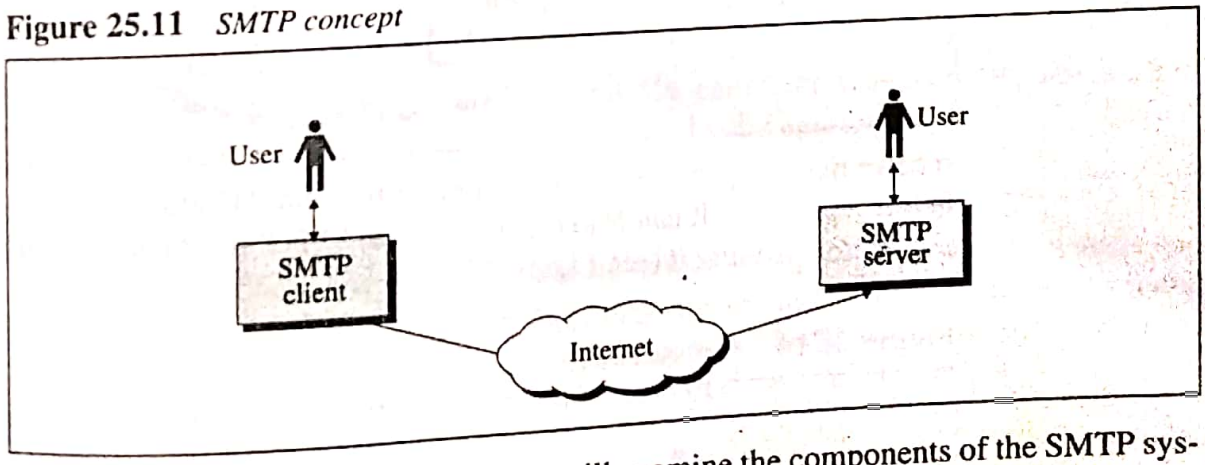
25.7 SIMPLE MAIL TRANSFER PROTOCOL (SMTP)

One of the most popular network services is **electronic mail (e-mail).** The TCP/IP protocol that supports electronic mail on the Internet is called **Simple Mail Transfer Protocol (SMTP).** It is a system for sending messages to other computer users based on e-mail addresses. SMTP provides for mail exchange between users on the same or different computers and supports:

- Sending a single message to one or more recipients.
- Sending messages that include text, voice, video, or graphics.
- Sending messages to users on networks outside the Internet.

Figure 25.11 shows the basic idea.

Figure 25.11 *SMTP concept*



Starting with this simple figure, we will examine the components of the SMTP system, gradually adding complexity. Let us begin by breaking down both the SMTP client and server into two components: **user agent (UA)** and **mail transfer agent (MTA).**

The UA prepares the message, creates the envelope, and puts the message in the envelope. The MTA transfers the mail across the Internet. Figure 25.12 shows the previous figure with the addition of these two components.

SMTP protocol allows a more complex system than the one shown. Relaying could be involved. Instead of just one MTA at the sender site and one at the receiving site, other MTAs, acting either as client or server, can relay the mail (see Figure 25.13).

The relaying system allows sites that do not use the TCP/IP protocol suite to send e-mail to users on other sites that may or may not use the TCP/IP protocol suite. This is accomplished through the use of a **mail gateway,** which is a relay MTA that can

Figure 25.12 *UAs and MTAs*

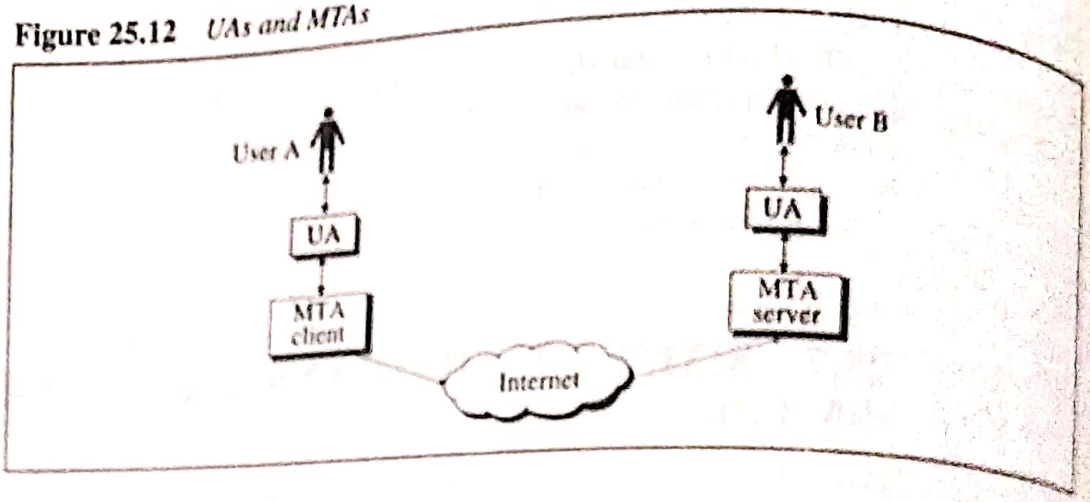
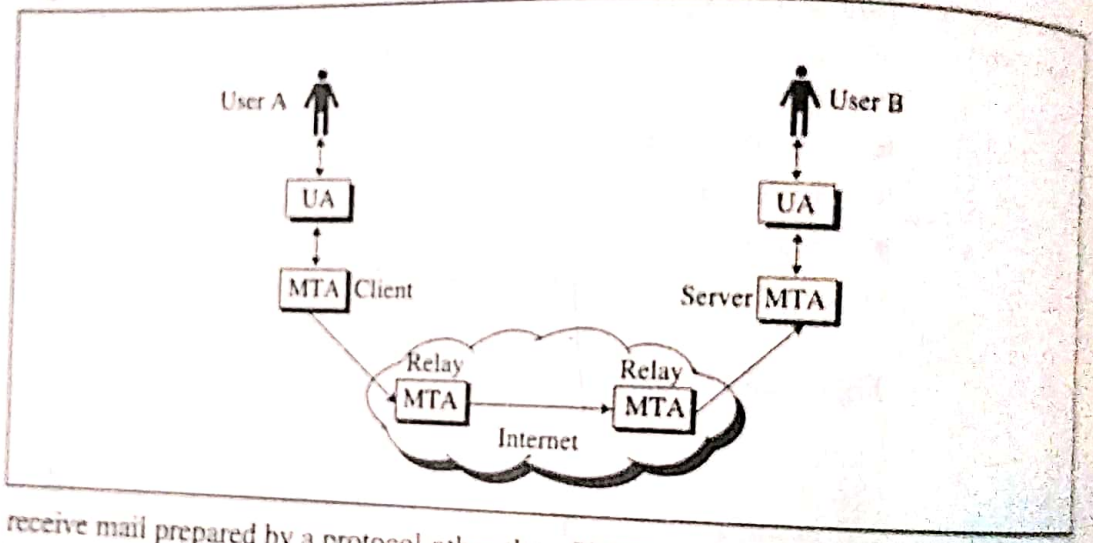
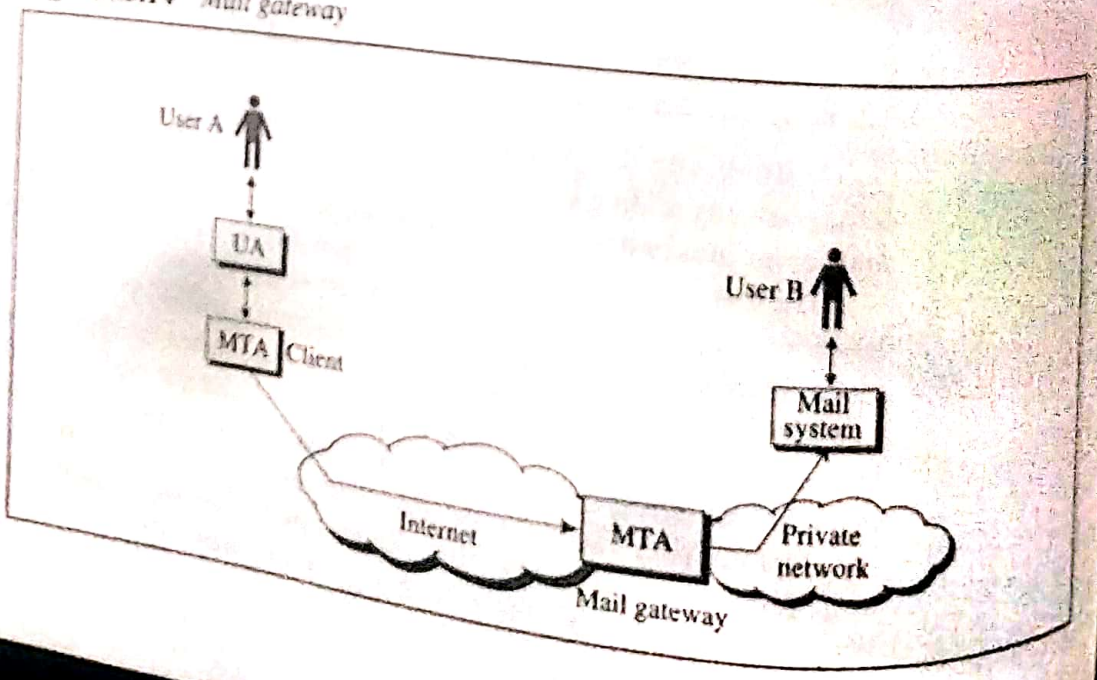


Figure 25.13 *Relay MTAs*



receive mail prepared by a protocol other than SMTP and transform it to SMTP format before sending it. It can also receive mail in SMTP format and change it to another format before sending it (see Figure 25.14).

Figure 25.14 *Mail gateway*



User Agent (UA)

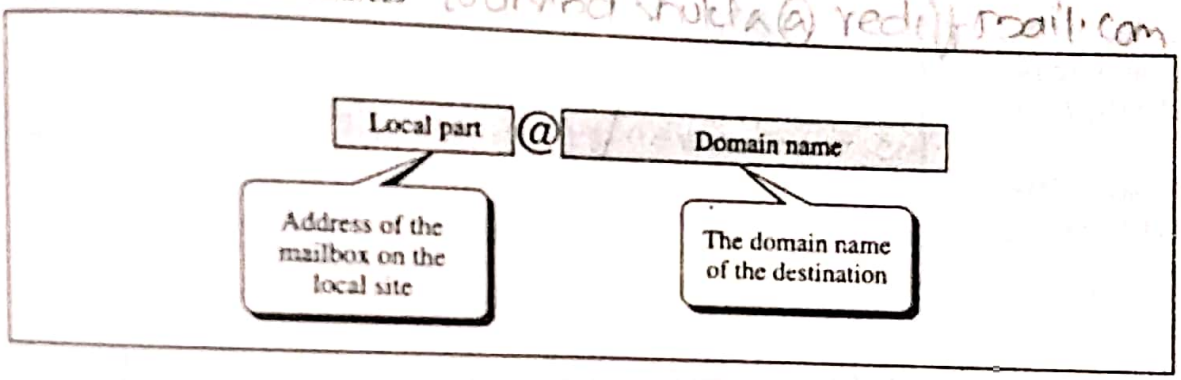
A user agent is defined in SMTP, but the implementation details are not. The UA is normally a program used to send and receive mail. Popular user agent programs are MH, Berkeley Mail, Elm, Zmail, and Mush.

Some user agents have an extra user interface that allows window-type interactions with the system.

Addresses

To deliver mail, a mail handling system must use a unique addressing system. The addressing system used by SMTP consists of two parts: a *local part* and a *domain name*, separated by an @ sign (see Figure 25.15).

Figure 25.15 *E-mail address*



Local Part

The local part defines the name of a special file, called the user mailbox, where all of the mail received for a user is stored for retrieval by the user agent.

Domain Name

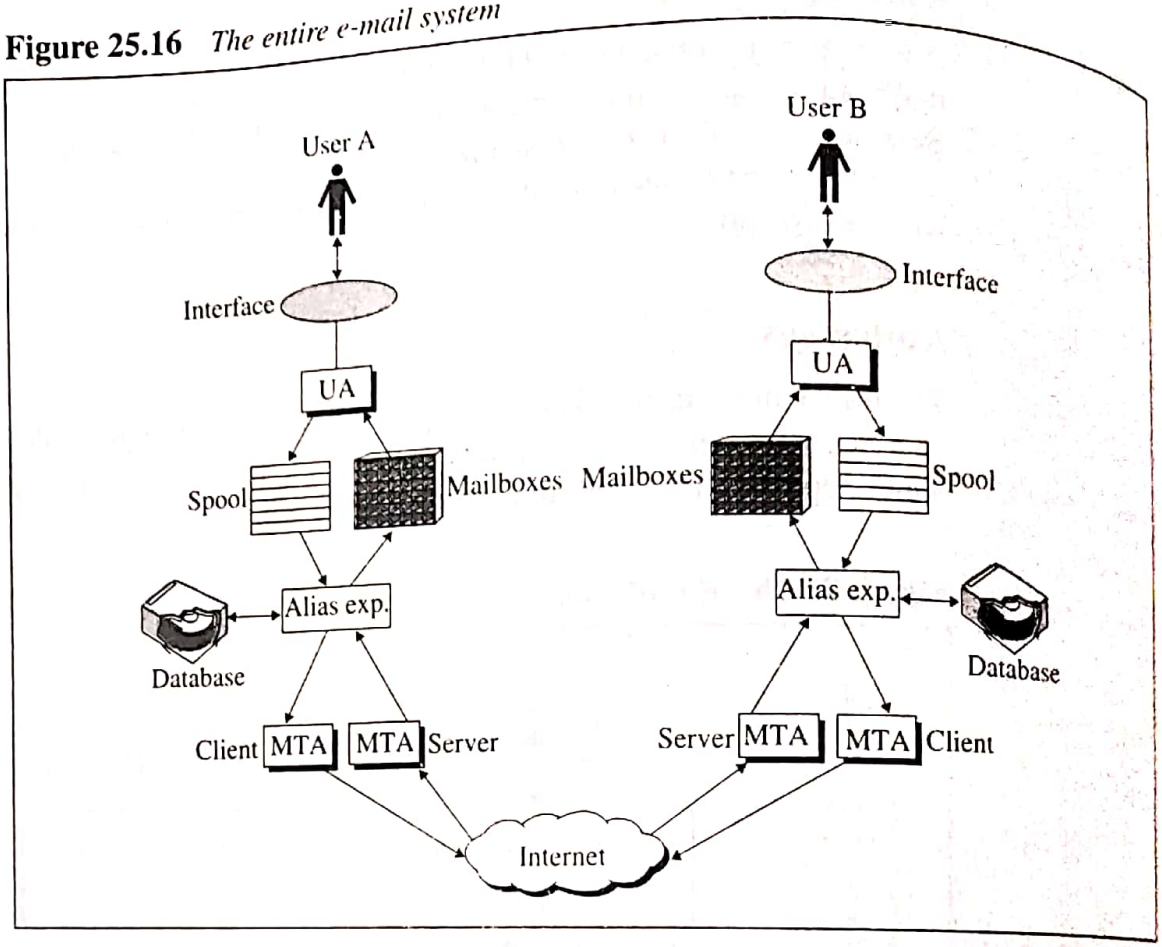
The second part of the address is the domain name. An organization usually selects one or more hosts to receive and send e-mail; they are sometimes called *mail exchangers*. The domain name assigned to each mail exchanger either comes from the DNS database or is a logical name (for example, the name of the organization).

Mail Transfer Agent (MTA)

The actual mail transfer is done through mail transfer agents (MTAs). To send mail, a system must have a client MTA, and to receive mail, a system must have a server MTA. Although SMTP does not define a specific MTA, Sendmail is a commonly used UNIX system MTA.

SMTP simply defines how commands and responses must be sent back and forth. Each network is free to choose a software package for implementation. Figure 25.16 illustrates the process of sending and receiving e-mail as described above. For a computer to be able to send and receive mail using SMTP, it must have most of the entities (the user interface is not necessary) defined in the figure. The user interface is a component that creates a user-friendly environment.

Figure 25.16 *The entire e-mail system*



Multipurpose Internet Mail Extensions (MIME)

SMTP is a simple mail transfer protocol. Its simplicity, however, comes with a price. SMTP can send messages only in NVT seven-bit ASCII format. In other words, it has some limitations. For example, it cannot be used for languages that are not supported by seven-bit ASCII characters (such as French, German, Hebrew, Russian, Chinese, and Japanese). Also, it cannot be used to send binary files or to send video or audio data.

Multipurpose Internet Mail Extension (MIME) is a supplementary protocol that allows non-ASCII data to be sent through SMTP. MIME is not a mail protocol and cannot replace SMTP; it is only an extension to SMTP.

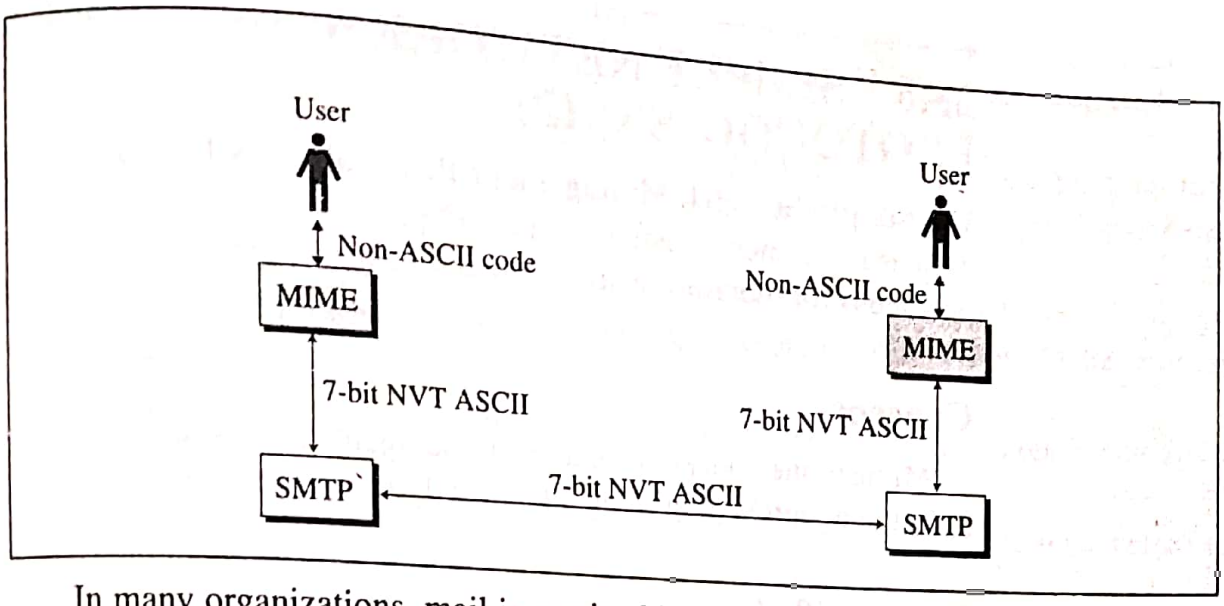
MIME transforms non-ASCII data at the sender site to NVT ASCII data and delivers it to the client SMTP to be sent through the Internet. The server SMTP at the receiving side receives the NVT ASCII data and delivers it to MIME to be transformed back to the original data.

We can think of MIME as a set of software functions that transform non-ASCII data to ASCII data and vice versa (see Figure 25.17).

Post Office Protocol (POP)

SMTP expects the destination host, the mail server receiving the mail, to be on-line all the time; otherwise, a TCP connection cannot be established. For this reason, it is not practical to establish an SMTP session with a desktop computer because desktop computers are usually powered down at the end of the day.

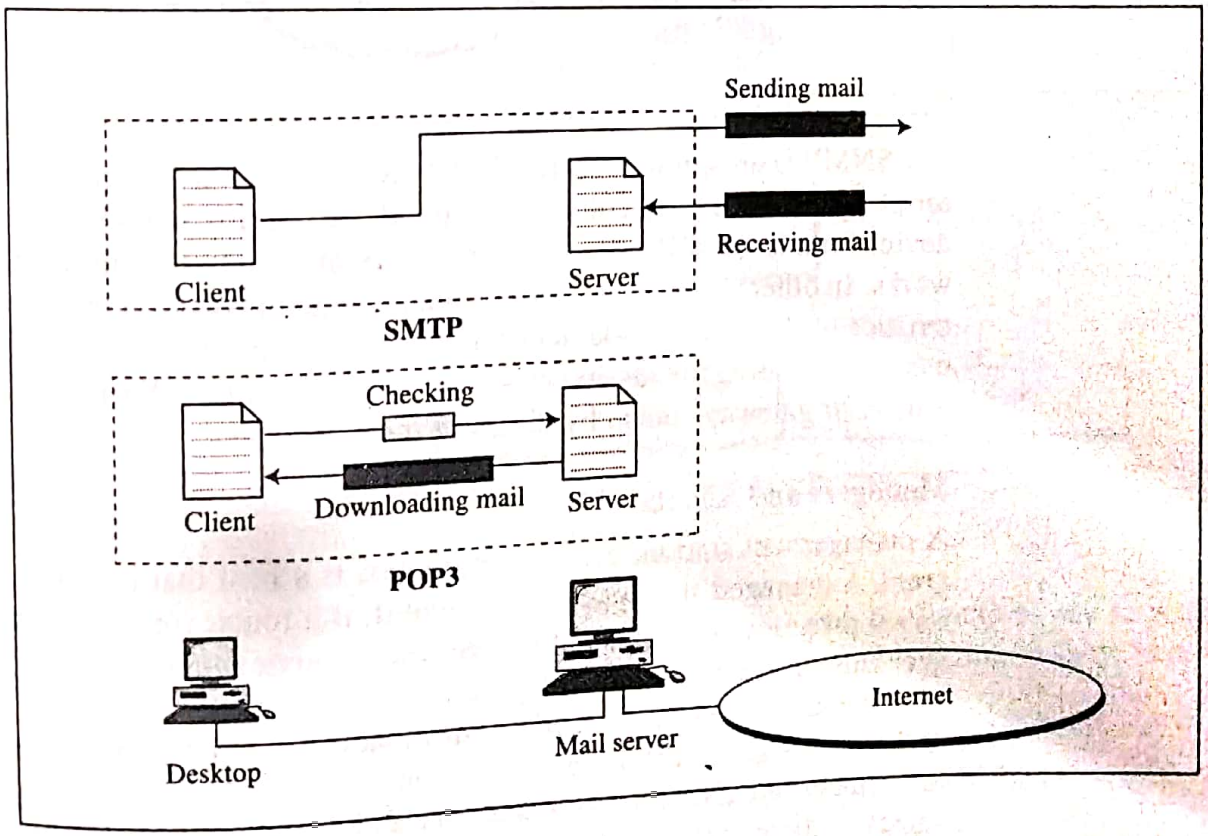
Figure 25.17 *MIME*



In many organizations, mail is received by an SMTP server that is always on-line. This SMTP server provides a mail-drop service. The server receives the mail on behalf of every host in the organization. Workstations interact with the SMTP host to retrieve messages by using a client–server protocol such as **Post Office Protocol (POP)**, version 3 (POP3).

Although POP3 is used to download messages from the server, the SMTP client is still needed on the desktop to forward messages from the workstation user to its SMTP mail server (see Figure 25.18).

Figure 25.18 *POP3 and SMTP*



Trap The Trap message is sent from the agent to the manager to report an event. For example, if the agent is rebooted, it informs the manager and reports the time of rebooting.

25.9 HYPERTEXT TRANSFER PROTOCOL (HTTP)

The **Hypertext Transfer Protocol (HTTP)** is a protocol used mainly to access data on the World Wide Web (see next section). The protocol transfers data in the form of plain text, hypertext, audio, video, and so on. However, it is called the hypertext transfer protocol because its efficiency allows its use in a hypertext environment where there are rapid jumps from one document to another.

HTTP functions like a combination of FTP and SMTP. It is similar to FTP because it transfers files and uses the services of TCP. However, it is much simpler than FTP because it uses only one TCP connection. There is not a separate control connection; only data are transferred between the client and the server.

HTTP is like SMTP because the data transferred between the client and the server look like SMTP messages. In addition, the format of the messages is controlled by MIME-like headers. However, HTTP differs from SMTP in the way the messages are sent from the client to the server and from the server to the client. Unlike SMTP, the HTTP messages are not destined to be read by humans; they are read and interpreted by the HTTP server and HTTP client (**browser**). SMTP messages are stored and forwarded, but HTTP messages are delivered immediately.

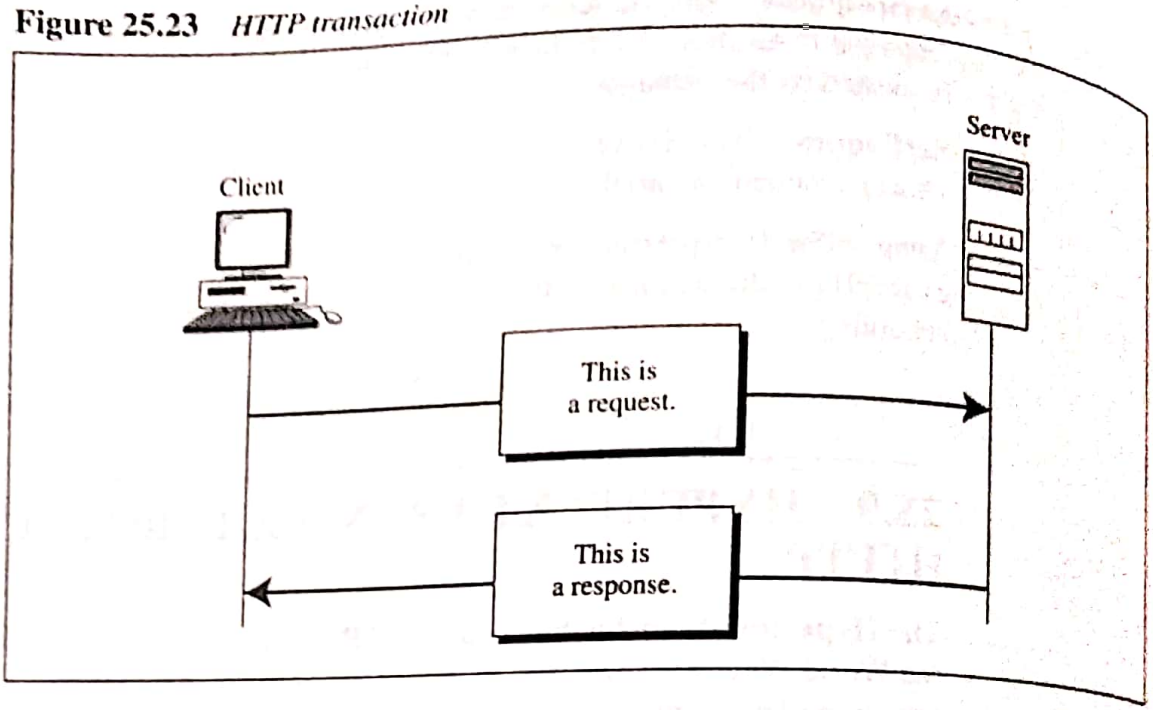
The idea of HTTP is very simple. A client sends a request, which looks like mail, to the server. The server sends the response, which looks like a mail reply, to the client. The request and response messages carry data in the form of a letter with MIME-like format.

The commands from the client to the server are embedded in a letterlike request message. The contents of the requested file or other information are embedded in a letterlike response message.

HTTP Transaction

Figure 25.23 illustrates the HTTP transaction between the client and server. The client initializes the transaction by sending a request message. The server replies by sending a response.

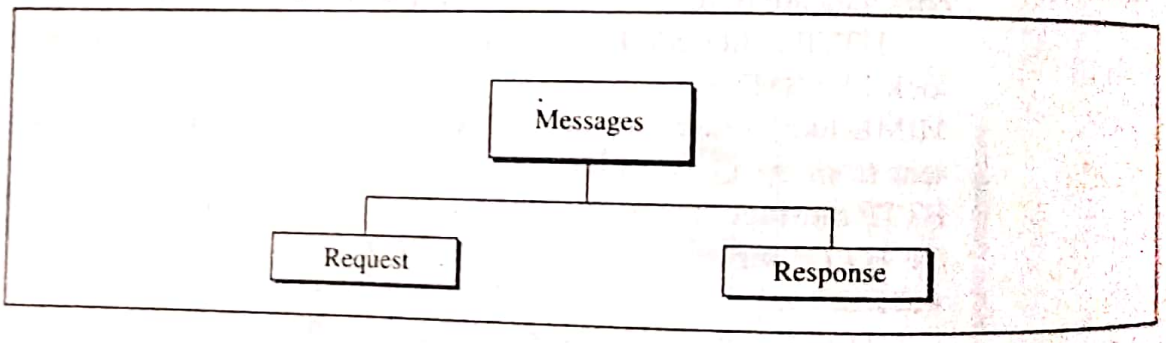
Figure 25.23 *HTTP transaction*



Messages

There are two general types of HTTP messages, shown in Figure 25.24: request and response. Both message types follow almost the same format.

Figure 25.24 *Message categories*



Request Messages

A request message consists of a request line, headers, and sometimes a body. See Figure 25.25.

Response Message

A response message consists of a status line, headers, and sometimes a body. See Figure 25.26.

Uniform Resource Locator (URL)

A client that wants to access a document needs an address. To facilitate the access of documents distributed throughout the world, HTTP uses the concept of locators. The **uniform resource locator (URL)** is a standard for specifying any kind of information

Figure 25.25 *Request message*

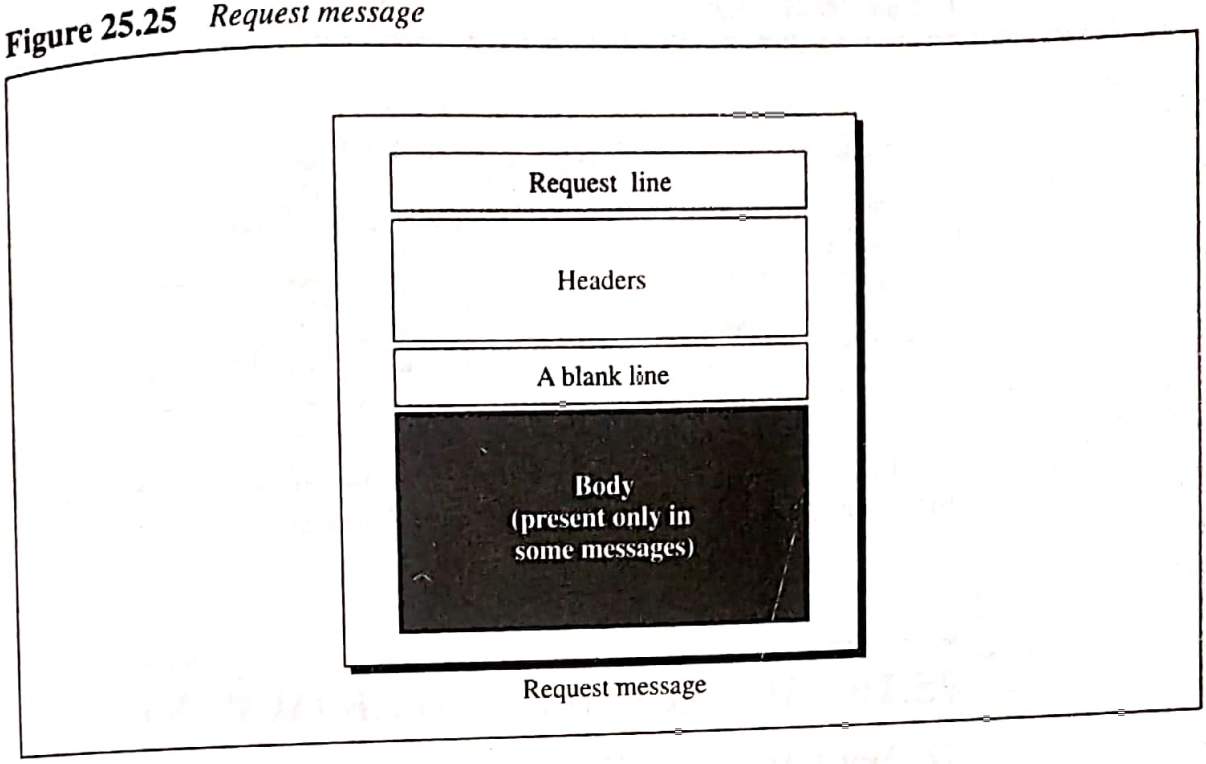
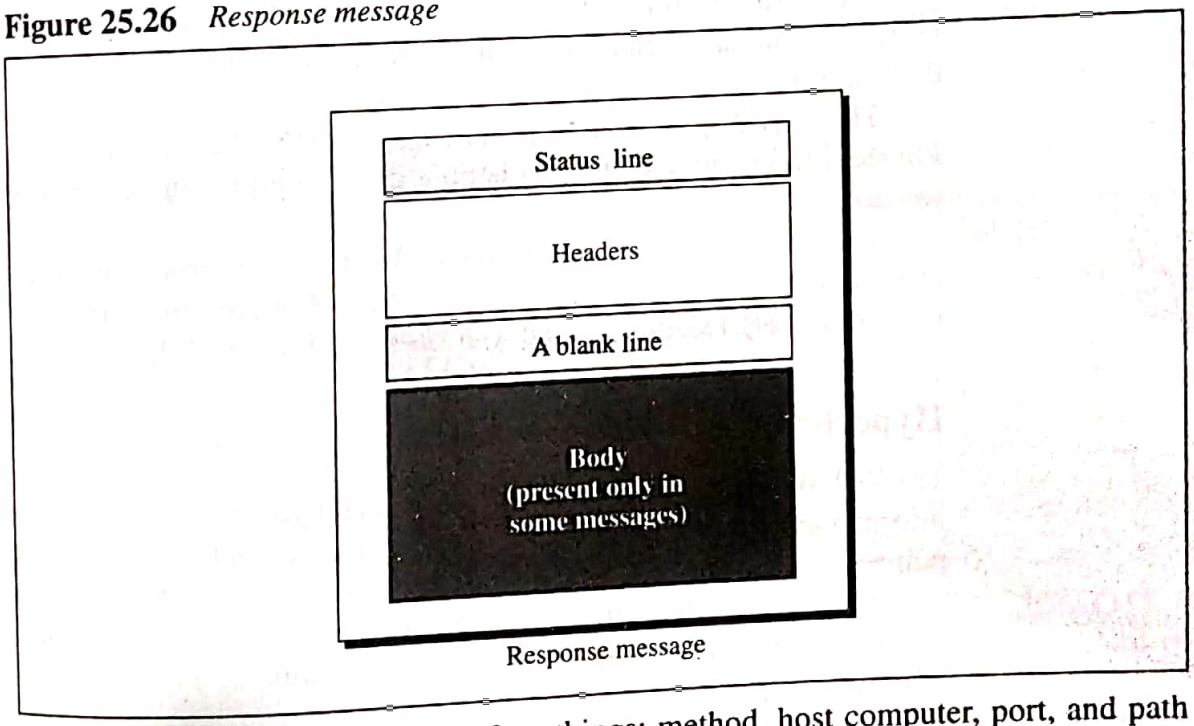


Figure 25.26 *Response message*

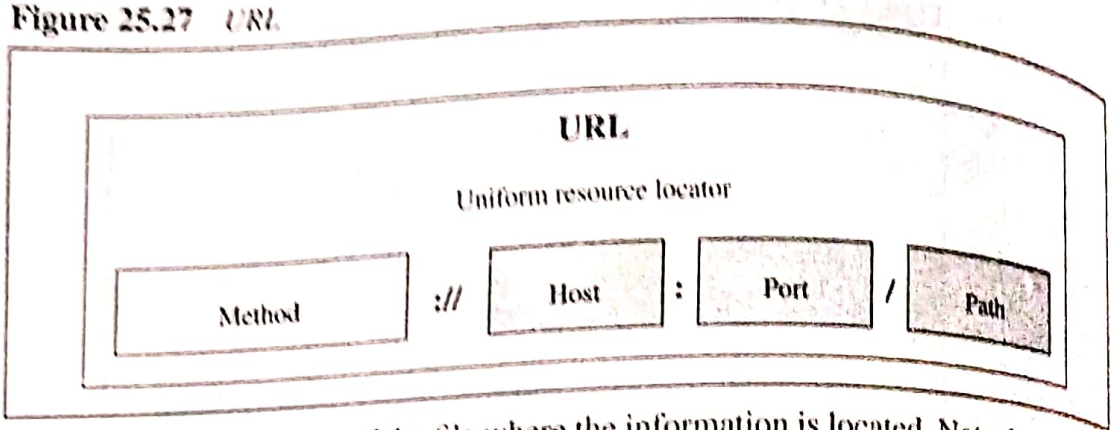


on the Internet. The URL defines four things: method, host computer, port, and path (see Figure 25.27).

The *method* is the protocol used to retrieve the document, for example HTTP. The *host* is the computer where the information is located, although the name of the computer can be an alias. Web pages are usually stored in computers, and computers are given alias names that usually begin with the characters "www." This is not mandatory, however, as the host can be any name given to the computer that hosts the web page.

The URL optionally can contain the port number of the server. If the *port* is included, it should be inserted between the host and the path, and it should be separated from the host by a colon.

Figure 25.27 *URL*



Path is the pathname of the file where the information is located. Note that the path can itself contain slashes that, in the UNIX operating system, separate the directories from the subdirectories and files.

25.10 WORLD WIDE WEB (WWW)

The **World Wide Web (WWW)**, or the **web**, is a repository of information spread all over the world and linked together. The WWW has a unique combination of flexibility, portability, and user-friendly features that distinguish it from other services provided by the Internet.

The WWW project was initiated by CERN (European Laboratory for Particle Physics) to create a system to handle distributed resources necessary for scientific research.

The WWW today is a distributed client–server service, in which a client using a browser can access a service using a server. However, the service provided is distributed over many locations called *web sites* (see Figure 25.28).

Hypertext and Hypermedia

The WWW uses the concept of hypertext and hypermedia. In a hypertext environment, information is stored in a set of documents that are linked together using the concept of pointers. An item can be associated with another document using a pointer. The reader who is browsing through the document can move to other documents by choosing (clicking) the items that are linked to other documents. Figure 25.29 shows the concept of hypertext.

Whereas hypertext documents contain only text, hypermedia documents can contain pictures, graphics, and sound.

A unit of hypertext or hypermedia available on the Web is called a *page*. The main page for an organization or an individual is known as a **homepage.**

Browser Architecture

A variety of vendors offer commercial browsers that interpret and display a web document, and all of them use nearly the same architecture. Each browser usually consists

Figure 25.28 *Distributed services*

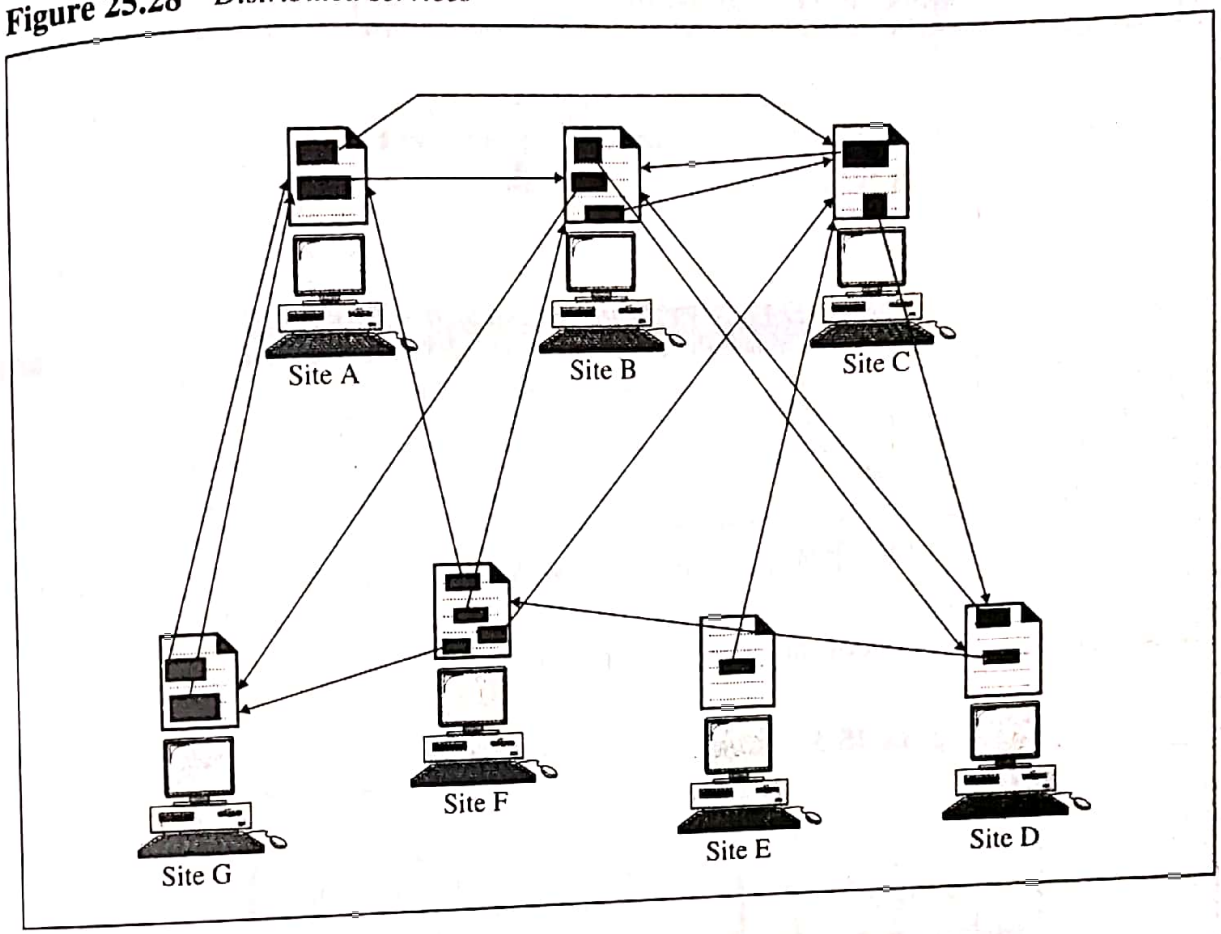
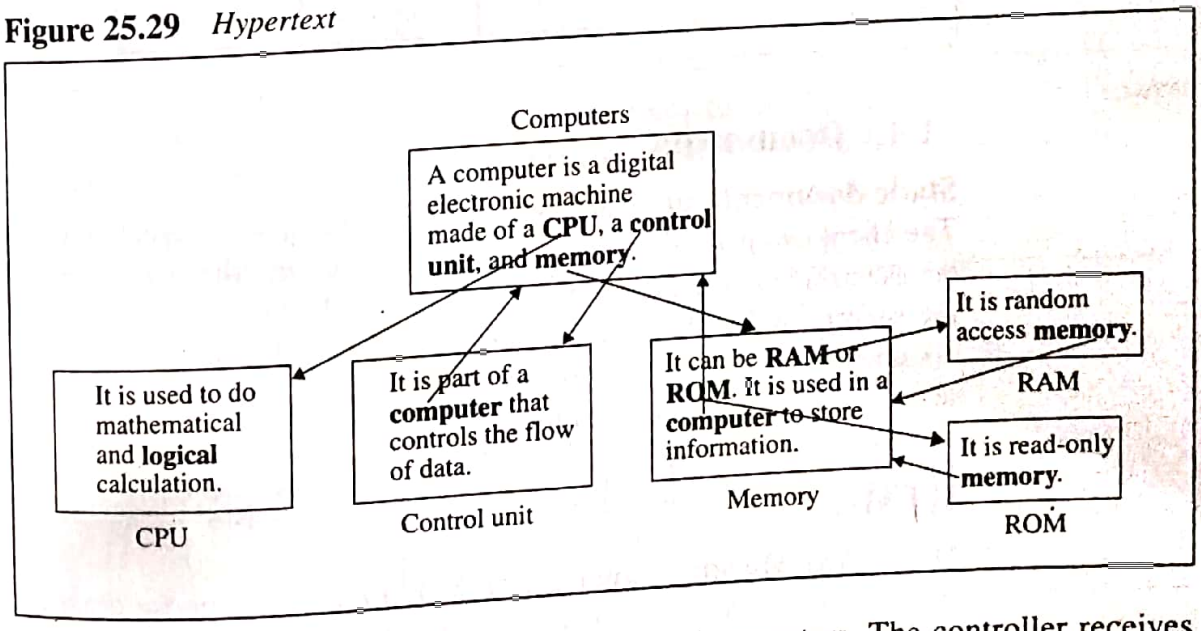
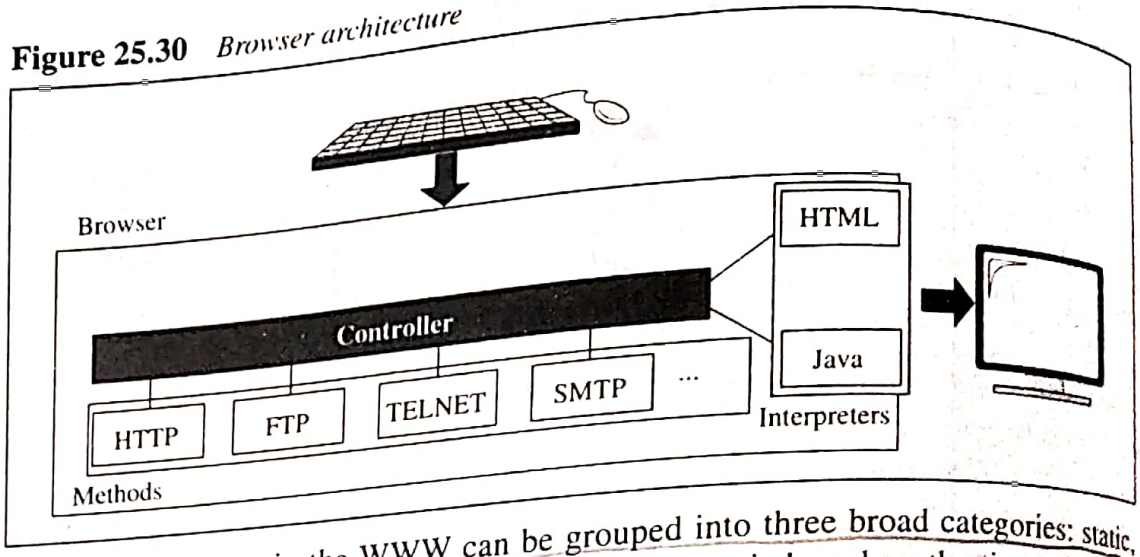


Figure 25.29 *Hypertext*



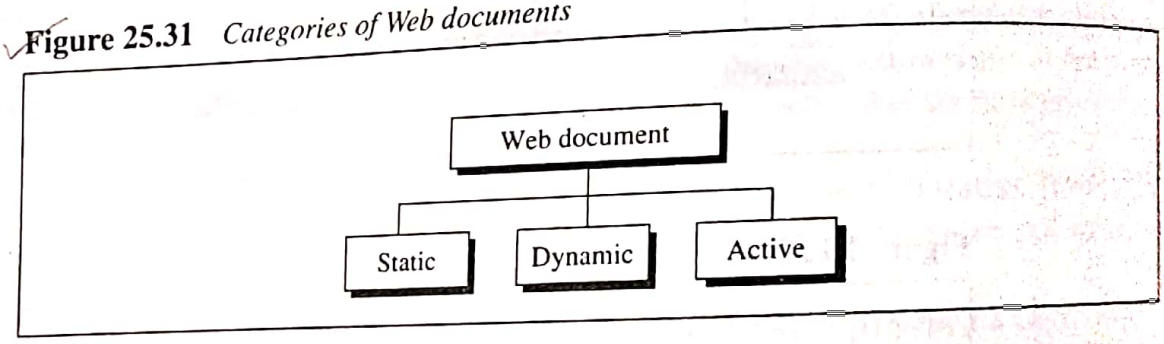
of three parts: a controller, client programs, and interpreters. The controller receives input from the keyboard or the mouse and uses the client programs to access the document. After the document has been accessed, the controller uses one of the interpreters to display the document on the screen. The client programs can be one of the methods (protocols) described previously such as HTTP, FTP, or TELNET. The interpreter can be HTML or Java, depending on the type of document (see Figure 25.30).

Figure 25.30 *Browser architecture*



The documents in the WWW can be grouped into three broad categories: static, dynamic, and active (see Figure 25.31). The category is based on the time when the contents of the document are determined.

Figure 25.31 *Categories of Web documents*



Static Documents

Static documents are fixed-content documents that are created and stored in a server. The client can get only a copy of the document. In other words, the contents of the file are determined when the file is created, not when it is used. Of course, the contents in the server can be changed, but the user cannot change it. When a client accesses the document, a copy of the document is sent. The user can then use a browsing program to display the document (see Figure 25.32).

HTML

HyperText Markup Language (HTML) is a language for creating web pages. The term *markup language* comes from the book publishing industry. Before a book is typeset and printed, a copy editor reads the manuscript and puts a lot of marks on it. These marks tell the designer how to format the text. For example, if the copy editor wants part of a line to be printed in boldface, he or she draws a wavy line under that part. In the same way, data for a web page are formatted for interpretation by a browser.

Let us explain the idea with an example. To make part of a text displayed in boldface with HTML, we must include the beginning and ending boldface tags (marks) in the text, as shown in Figure 25.33.

Figure 25.32 *Static document*

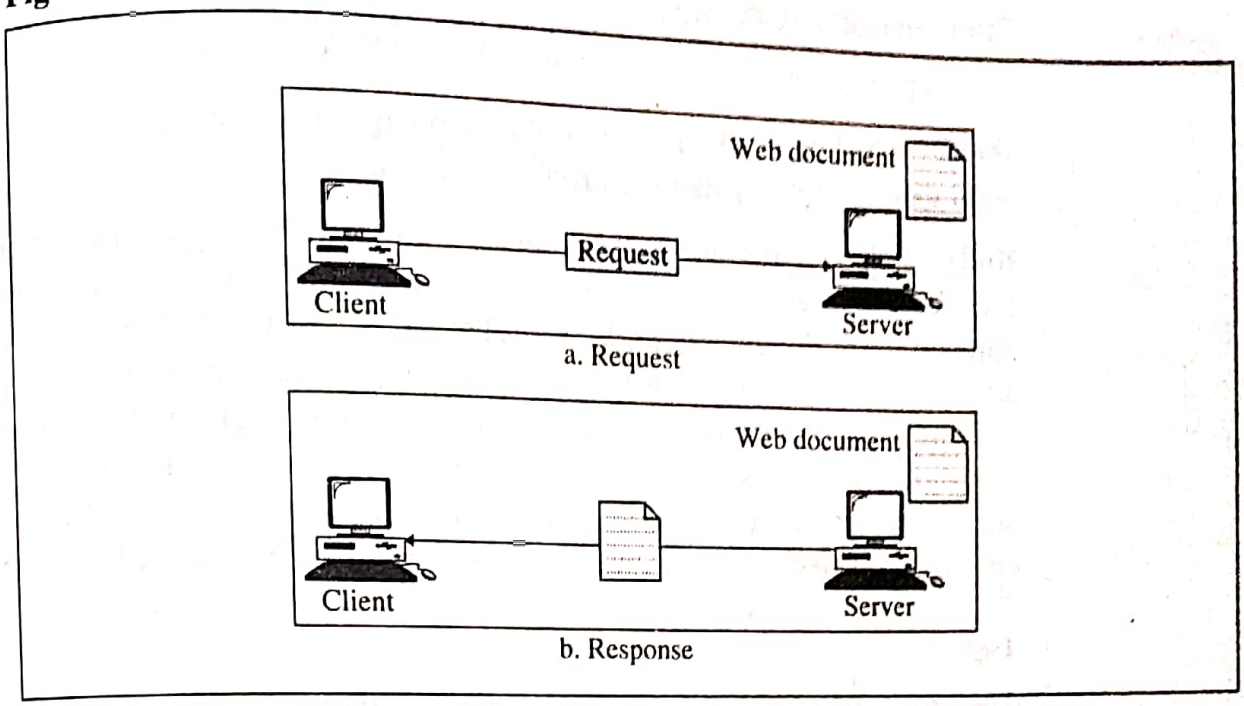
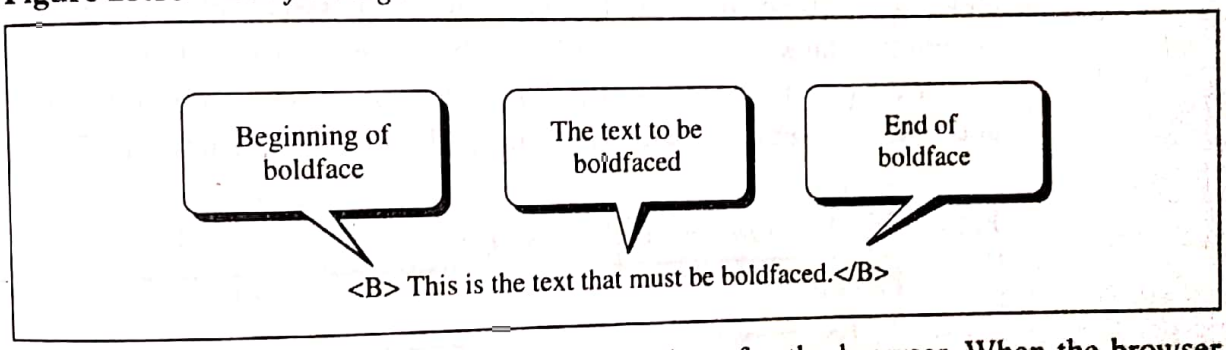
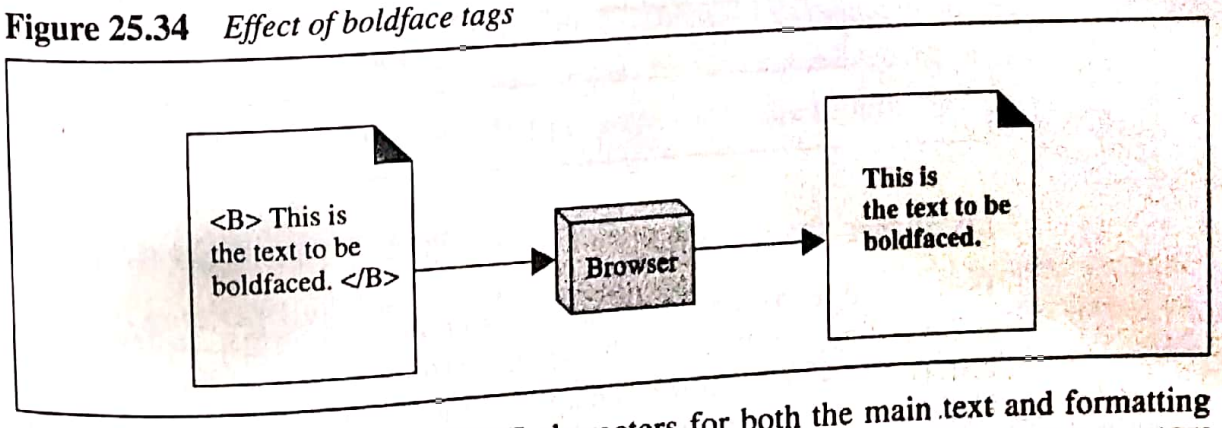


Figure 25.33 *Boldface tags*



The two tags <B> and </B> are instructions for the browser. When the browser sees these two marks, it knows that the text must be boldfaced (see Figure 25.34).

Figure 25.34 *Effect of boldface tags*



HTML lets us use only ASCII characters for both the main text and formatting instructions. In this way, every computer can receive the whole document as an ASCII document. The main text is the data, and the formatting instructions can be used by the browser to format the data.

Structure of a Web Page

A web page is made up of two parts: the head and the body.

Head The head is the first part of a web page. The head contains the title of the page and other parameters that the browser will use.

Body The actual contents of a page are in the body, which includes the text and the tags. Whereas the text is the actual information contained in a page, the tags define the appearance of the document. Every HTML tag is a name followed by an optional list of attributes, all enclosed between less than and greater than brackets (< and >).

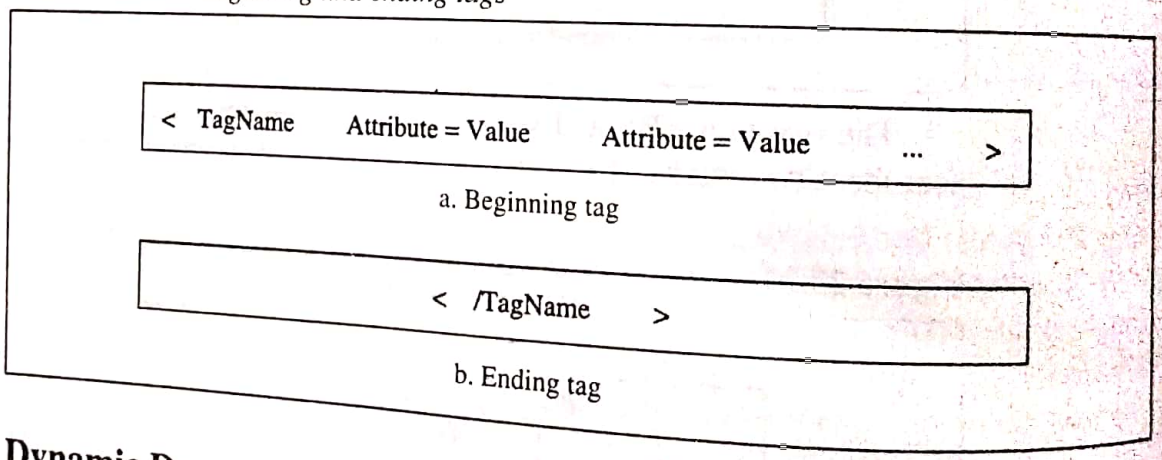
An attribute, if present, is followed by an equal sign and the value of the attribute. Some tags can be used alone; some must be used in pairs. Those that are used in pairs are called *starting* and *ending* tags. The starting tag can have attributes and values. The ending tag cannot have attributes or values but must have a slash before the name.

Tags

The browser makes a decision about the structure of the text based on the tags, which are marks that are embedded into the text. A tag is enclosed in two brackets (< and >) and usually comes in pairs. The beginning tag starts with the name of the tag, and the ending tag starts with a slash followed by the name of the tag.

A tag can have a list of attributes, each of which can be followed by an equal sign and a value associated with the attribute. Figure 25.35 shows the format of a tag.

Figure 25.35 *Beginning and ending tags*

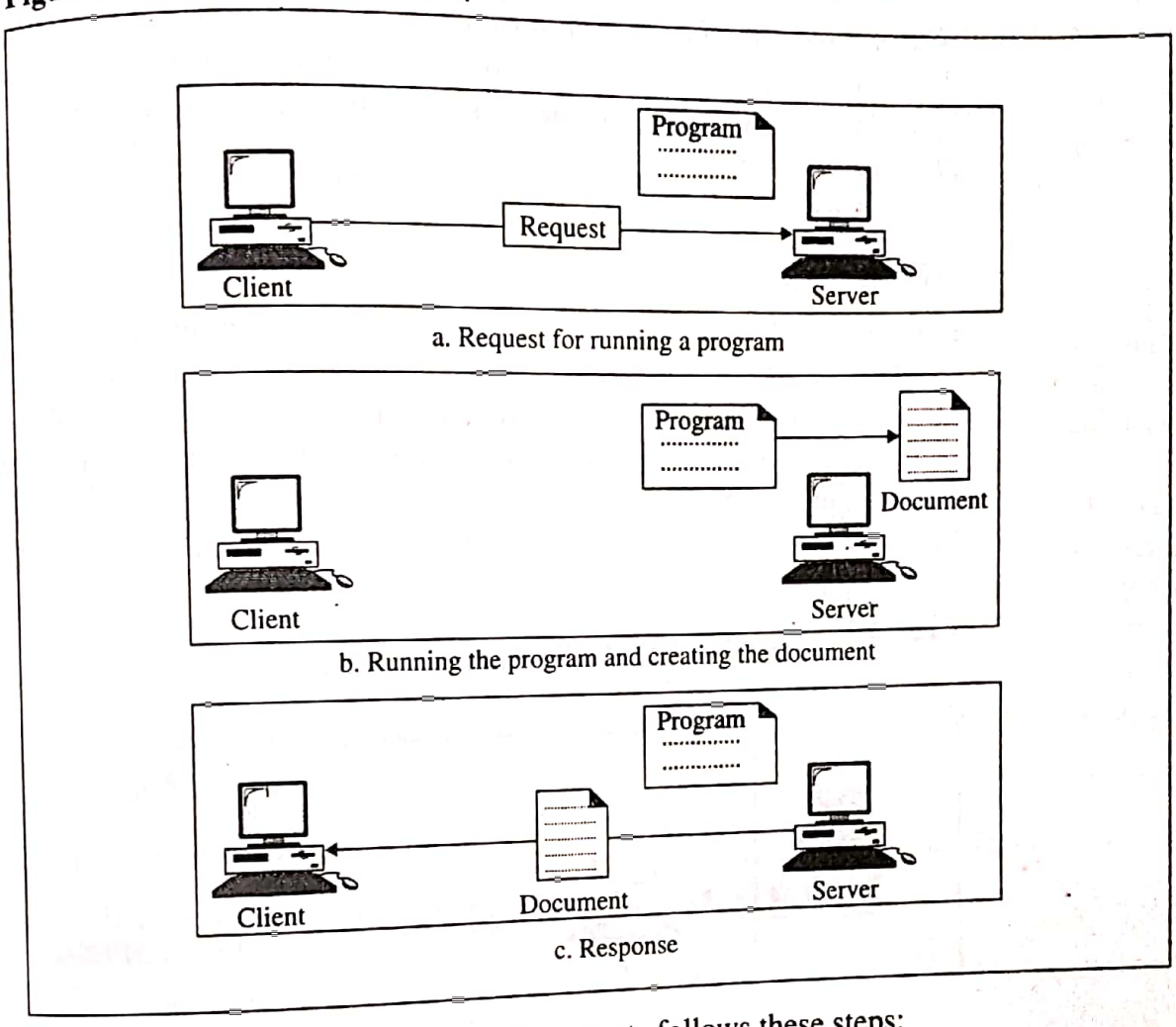


Dynamic Documents

Dynamic documents do not exist in a predefined format. Instead, a dynamic document is created by a web server whenever a browser requests the document. When a request arrives, the web server runs an application program that creates the dynamic document. The server returns the output of the program as a response to the browser that requested the document. Because a fresh document is created for each request, the contents of a dynamic document can vary from one request to another. A very simple example of a dynamic document is getting the time and date from the server. Time and date are kinds of information that are dynamic in that they change from moment to moment. The client can request that the server run a program such as the date program in UNIX and

send the result of the program to the client. Figure 25.36 illustrates the steps in sending and responding to a dynamic document.

Figure 25.36 *Dynamic document*



A server that handles dynamic documents follows these steps:

1. The server examines the URL to find if it defines a dynamic document.
2. If the URL defines a dynamic document, the server executes the program.
3. The server sends the output of the program to the client (browser).

Common Gateway Interface (CGI)

Common gateway interface (CGI) is a technology that creates and handles dynamic documents. CGI is a set of standards that defines how a dynamic document should be written, how input data should be supplied to the program, and how the output result should be used.

CGI is not a new language; instead, it allows programmers to use any of several languages such as C, C++, Bourne Shell, Korn Shell, C Shell, Tcl, or Perl. The only thing that CGI defines is a set of rules and terms that the programmer should follow.

The use of *common* in CGI indicates that the standard defines a set of rules that are common to any language or platform. The term *gateway* here means that a CGI program is a gateway that can be used to access other resources such as databases, graphic

20. Compare and contrast the three types of WWW documents.
21. What is the purpose of HTML?
22. What is the relationship between CGI and dynamic documents?
23. What is Java?

Multiple Choice Questions

24. _____ can request a service.
    a. A socket interface
    b. A port
    c. A client
    d. A server
25. The client program is _____ because it terminates after it has been served.
    a. active
    b. passive
    c. finite
    d. infinite
26. The server program is _____ because it is always available, waiting for a client request.
    a. active
    b. passive
    c. finite
    d. infinite

27. The TCP/IP application layer corresponds to the OSI model's _____ layers.
    a. physical, data link, and network
    b. transport and network
    c. session and transport
    d. session, presentation, and application
28. To find the IP address of a host when the domain name is known, the _____ can be used.
    a. inverse domain
    b. generic domains
    c. country domains
    d. b or c
29. Remote login can involve _____.
    a. NVT
    b. TELNET
    c. TCP/IP
    d. all of the above
30. The _____ at the remote site sends received characters to the operating system.
    a. terminal driver
    b. pseudoterminal driver
    c. TELNET client
    d. TELNET server
31. The _____ translates local characters into NVT form.
    a. terminal driver
    b. pseudoterminal driver
    c. TELNET client
    d. TELNET server
32. The _____ translates NVT characters into a form acceptable by the operating system.
    a. terminal driver
    b. pseudoterminal driver
    c. TELNET client
    d. TELNET server
33. Which of the following is true?
    a. FTP allows systems with different directory structures to transfer files.
    b. FTP allows a system using ASCII and a system using EBCDIC to transfer files.
    c. FTP allows a PC and a SUN workstation to transfer files.
    d. all of the above
34. During an FTP session, the control connection is opened _____.
    a. exactly once

b. exactly twice
c. as many times as necessary
d. all of the above

35. During an FTP session, the data connection is opened _____.
a. exactly once
b. exactly twice
c. as many times as necessary
d. all of the above

36. The purpose of the UA is _____.
a. message preparation
b. envelope creation
c. transferal of messages across the Internet
d. a and b

37. The purpose of the MTA is _____.
a. message preparation
b. envelope creation
c. transferal of messages across the Internet
d. a and b

38. When a message is sent using SMTP, _____ UA(s) are involved.
a. only one
b. only two
c. only three
d. at least two

39. E-mail cannot be sent _____.
a. if the sending site does not use TCP/IP
b. if the receiving site does not use TCP/IP
c. through private networks
d. none of the above

40. A _____ can transform non-SMTP mail to SMTP format and vice versa.
a. mail spool
b. mail gateway
c. mail file
d. mail exchanger

41. In the mail address mackenzie@pit.arc.nasa.gov, what is the domain name?
a. mackenzie
b. pit.arc.nasa.gov
c. mackenzie@pit.arc.nasa.gov
d. a and b

42. MIME allows _____ data to be sent through SMTP.
a. audio

b. non-ASCII data
c. image
d. all of the above

43. Which of the following is associated with SNMP?
a. MIB
b. SMI
c. BER
d. all of the above

44. _____ runs the SNMP client program; _____ runs the SNMP server program.
a. A manager; a manager
b. An agent; an agent
c. A manager; an agent
d. An agent; a manager

45. _____ names objects, defines the type of data that can be stored in an object, and encodes data for network transmission.
a. MIB
b. SMI
c. SNMP
d. ASN.1

46. Which of the following is a collection of objects to be managed?
a. MIB
b. SMI
c. SNMP
d. ASN.1

47. Which is a manager duty?
a. Retrieve the value of an object defined in an agent.
b. Store the value of an object defined in an agent.
c. Send an alarm message to the agent.
d. a and b

48. _____ specifies which data types are available for the MIB.
a. BER
b. SNMP
c. ASN.1
d. SMI

49. An SNMP agent can send _____ messages.
a. GetRequest
b. SetRequest
c. GetNextRequest
d. Trap

50. An SNMP manager can send _____ messages.
    a. GetRequest
    b. SetRequest
    c. GetNextRequest
    d. all of the above
51. An SNMP agent can send _____ messages.
    a. GetResponse
    b. GetRequest
    c. SetRequest
    d. GetNextRequest
52. HTTP has similarities to both _____ and _____.
    a. FTP; SNMP
    b. FTP; SMTP
    c. FTP; MTV
    d. FTP; URL
53. A request message always contains _____.
    a. a header and a body
    b. a request line and a header
    c. a status line, a header, and a body
    d. a status line and a header
54. What does the URL need to access a document?
    a. pathname
    b. host computer
    c. retrieval method
    d. all of the above
55. Which of the following is a retrieval method?
    a. HTTP
    b. FTP
    c. TELNET
    d. all of the above
56. Hypertext documents are linked through _____.
    a. DNS
    b. TELNET
    c. pointers
    d. homepages
57. Which of the following is not an interpreter?
    a. HTTP
    b. HTML
    c. CGI
    d. Java

58. What are the components of a browser?
    a. retrieval method, host computer, pathname
    b. controller, client program, interpreter
    c. hypertext, hypermedia, HTML
    d. all of the above
59. Which type of web document is run at the client site?
    a. static
    b. dynamic
    c. active
    d. all of the above
60. Which type of web document is created at the server site only when requested by a client?
    a. static
    b. dynamic
    c. active
    d. all of the above
61. Which type of web document is fixed-content and is created and stored at the server site?
    a. static
    b. dynamic
    c. active
    d. all of the above
62. A program can use _____ to write a CGI program.
    a. Bourne shell script
    b. Perl
    c. C
    d. any of the above
63. Which type of web document is transported from the server to the client in binary form?
    a. static
    b. dynamic
    c. active
    d. all of the above
64. _____ is used to enable the use of active documents.
    a. HTML
    b. CGI
    c. Java
    d. all of the above

65. Java is _____.
    a. a programming language
    b. a run-time environment
    c. a class library
    d. all of the above
66. An applet is _____ document application program.
    a. a static
    b. an active
    c. a passive
    d. a dynamic

Exercises

67. Which type of domain is used by your company or school (generic or country)?
68. Most companies prefer generic domains to country domains. Why?
69. What is the most common generic domain you have used in your e-mails?
70. Break your e-mail address into the domain name and the local name.
71. Do you know anyone using a country domain?
72. Have you ever used an inverse domain?
73. Have you ever used a TELNET command? Which one?
74. Can a TELNET command use an IP address?
75. When you want to send an e-mail, what happens if your computer cannot find a DNS server?
76. What user agent program does your system use?
77. Does your e-mail program use MIME?
78. Does your e-mail program use POP?
79. Use the get command in FTP and report the result.
80. Use the put command in FTP and report the result.
81. Find the list of user commands in FTP.
82. Do some research on anonymous FTP and discuss its function.
83. Have you ever created an e-mail list?
84. Do some research and find at least five different tags used in HTML.