Module 5 The Application Layer

The Domain Name System DNS

computers on a network can theoretically access resources such as web pages and email accounts using numerical network addresses (like IP addresses), this approach is not ideal for human users. IP addresses are difficult to remember and are not convenient for everyday use. For example, accessing a company's website using an address like: 128.111.24.41 would be inconvenient and impractical. Additionally, if the company moves its website to another server with a different IP address, every user would need to be informed of the new address.

To solve this issue, high-level, human-friendly names (called **domain names**) were introduced. These names allow users to refer to resources using familiar text labels rather than long numeric sequences.

Example:

www.cs.washington.edu

This name remains the same even if the underlying server's IP address changes. Since the network itself can only understand numerical addresses, a mechanism is required to translate readable names into machine-level IP addresses. This process is known as **name resolution** and is handled by the **Domain Name System (DNS)**.

What is mean by Domain Name Space

- **DNS (Domain Name System)** is a hierarchical and distributed naming system used on the Internet to translate human-readable domain names (such as **www.google.com**) into machine-readable IP addresses (like **142.250.190.78**).

- Managing a large and continuously changing collection of host names across the Internet is a complex task. To solve this, DNS (Domain Name System) uses a **hierarchical naming structure**, similar to how postal addressing works.

- For the Internet, the top of the naming hierarchy is managed by an organization called **ICANN** (Internet Corporation for Assigned Names and Numbers).

It organizes the global namespace into a hierarchy of domains. At the highest level, there are **top-level domains (TLDs)**. There are more than **250 TLDs**, such as:

- ✓ .com
- ✓ .edu
- ✓ .gov
- ✓ .org
- ✓ Country codes (e.g., .in, .uk, .jp)

Each top-level domain is divided into **subdomains**, and subdomains may be further divided, forming a tree-like structure.

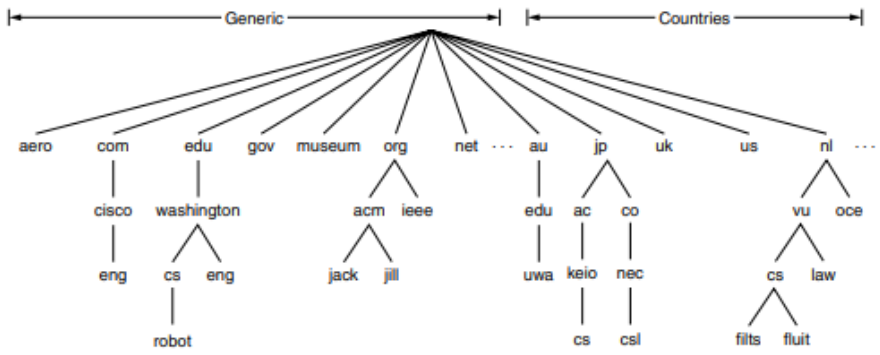


Figure 7-1. A portion of the Internet domain name space.

- Top-level domains are divided into two main categories:
  1. Generic Top-Level Domains (gTLDs)
  These include original Internet domains created in the 1980s (such as .com, .edu, .gov, .org, .net).

  Additional generic domains have since been introduced through applications submitted to ICANN.
  More gTLDs may continue to be added in the future
  2. Country Code Top-Level Domains (ccTLDs)
  Each country receives a unique two-letter code, based on ISO 3166.
- Examples:
  .in (India) .uk (United Kingdom) .jp (Japan)
  In 2010, internationalized domain names (IDNs) were introduced, allowing domains to be written in non-Latin scripts such as: Arabic, Cyrillic, Chinese and Other native languages.
- DNS assigns names to domains based on their position in a hierarchical naming tree. Each domain name identifies its location in the hierarchy by listing its components from the leaf up to the root, separated by periods (.), which are read as "dot". Example eng.cisco.com.

DNS Resource Records

- In DNS, every domain—whether it represents a single host or an entire top-level domain—contains a set of **resource records (RRs)**. These records collectively form the **DNS database**.

Purpose of Resource Records

- Resource records store information about a domain. The most common record stores the IP address of a host, but DNS supports many other record types as well.
- When a **resolver** queries DNS with a domain name (example: www.example.com), DNS returns the **resource records** associated with that name. The format is as follows: **Domain name, Time to live, Class, Type, Value.**
- **The Domain name tells the domain** to which this record applies. Normally, many records exist for each domain and each copy of the database holds information about multiple domains. This field is thus the primary search key used to satisfy queries. The order of the records in the database is not significant.
- The **Time to live** field gives an indication of how stable the record is. Information that is highly stable is assigned a large value, such as 86400 (the number of seconds in 1 day).

Information that is highly volatile is assigned a small value, such as 60 (1 minute).

- The third field of every resource record is the **Class**. For Internet information, it is always IN. For non-Internet information, other codes can be used, but in practice these are rarely seen.
- The **Type field** tells what kind of record this is. There are many kinds of DNS records.

Name Servers

- DNS namespace is logically divided into multiple **nonoverlapping zones**. Each zone is responsible for storing and maintaining the resource records for its portion of the DNS hierarchy.
- The placement of **zone boundaries** in the DNS hierarchy is determined by the **administrator of the zone**. The decision usually depends on:

  - ✓ How many name servers are needed?
  - ✓ Where those servers will be located
  - ✓ How administrative responsibilities are divided

- In the example shown: washington.edu is one zone. It includes eng.washington.edu. However, cs.washington.edu is a separate zone with its own name servers. This separation may occur because: Some departments (e.g., English) may not want to manage their own DNS server. Others (e.g., Computer Science) may have the expertise and need to manage their own zone.

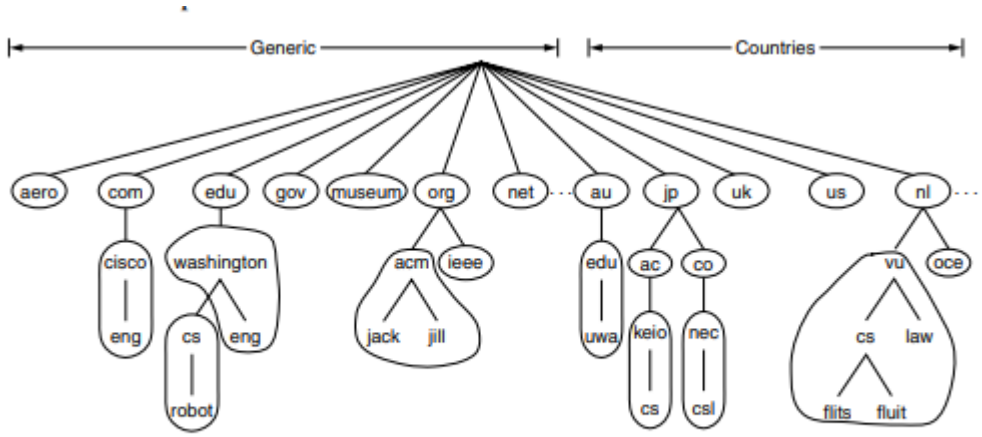


Figure 7-5. Part of the DNS name space divided into zones (which are circled).

- The act of mapping a domain name to an IP address is called **name resolution**.

How It Works:

1. A **resolver** (usually part of the OS or application) receives a query.
2. It forwards the query to a **local DNS name server**.
3. If the name falls within the zone managed by that server (e.g., top.cs.vu.nl belongs to cs.vu.nl), the server returns the answer.

- An **authoritative record** is one that comes from the authority that manages the record and is thus always correct. Authoritative records are in contrast to **cached records**, which may be out of date.

- When a user wants to find the IP address of a domain such as **robot.cs.washington.edu**, the DNS performs a step-by-step lookup across multiple name servers. If the local server has no cached information available, it follows a **remote resolution process**, as shown in Figure 7-6.

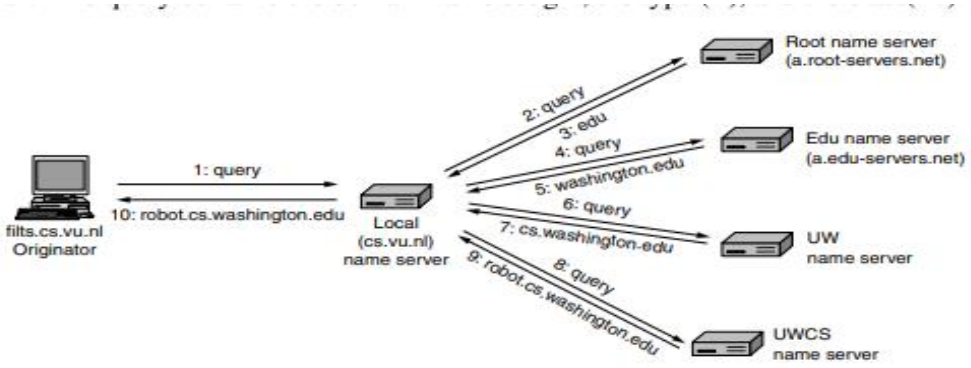


Figure 7-6. Example of a resolver looking up a remote name in 10 steps.

- **<u>Step-by-Step Resolution (Based on the Figure)</u>**

1. The user's computer (resolver) sends a query to the local name server (cs.vu.nl).
2. The local name server sends a query to a root name server.
3. The root name server responds with the name and IP of the .edu top-level domain (TLD) name server.
4. The local server sends the query to the edu name server.
5. The edu name server replies with the name server for washington.edu.
6. The local name server sends the request to the University of Washington (UW) name server.
7. UW responds with the name server for the cs.washington.edu department.
8. The local name server queries the UW Computer Science (UWCS) name server.
9. UWCS returns the final IP address of robot.cs.washington.edu.
10. The local server forwards the answer to the originating host.

- The name has now been resolved.
- **Caching in DNS:** Caching is used to speed up responses and reduce network load. All DNS responses, including partial answers, are stored temporarily Future queries for the same domain or related domains may be resolved much faster.
- Example benefits:
  - ✓ A second lookup for robot.cs.washington.edu is immediate.
  - ✓ A lookup for galah.cs.washington.edu goes directly to the authoritative UWCS server.

<u>Electronic Mail</u>

- **Electronic mail (commonly called e-mail)** is a method of creating, sending, receiving,

and storing digital messages using computer networks, especially the Internet. It allows users to exchange information quickly and efficiently without needing physical mail.

- It is a **digital communication system** that replaces traditional postal mail.
- Messages may include **text, images, files, hyperlinks, and multimedia attachments**.
- Email uses a **client-server model**, where messages are stored on email servers and accessed by users through email applications (e.g., Gmail, Outlook).
- **How It Works (Simple Flow):** - Sender → Email Server → Internet → Recipient's Email Server → Recipient

Architecture and Services

- Electronic mail systems are organized into different components that work together to send, receive, and manage messages over a network. The general architecture of an email system is illustrated in **Figure 7-7** and consists of two main subsystems:
  1. **User Agents (UA)**
  2. **Message Transfer Agents (MTA)**

1. User Agents (UA)

A **user agent** is the software that users interact with directly to manage their email. Examples include **Gmail, Outlook, Thunderbird**, and mobile mail apps.

Functions of a User Agent:

- Compose new messages
- Read received messages
- Reply to, forward, delete, and organize mail
- Search mail folders and archives
- Provide either a graphical interface (GUI) or a text-based/command-line interface

The process of sending a message from the user agent into the mail system for delivery is known as **mail submission**.

2. Message Transfer Agents (MTA)

A **Message Transfer Agent**, also called a **mail server**, is responsible for transferring email messages across the network from the sender to the recipient.
Examples include servers using **SMTP (Simple Mail Transfer Protocol)**.

The MTA forwards messages through one or more intermediate servers until they reach the destination mail server, where the recipient can retrieve them using a user agent.

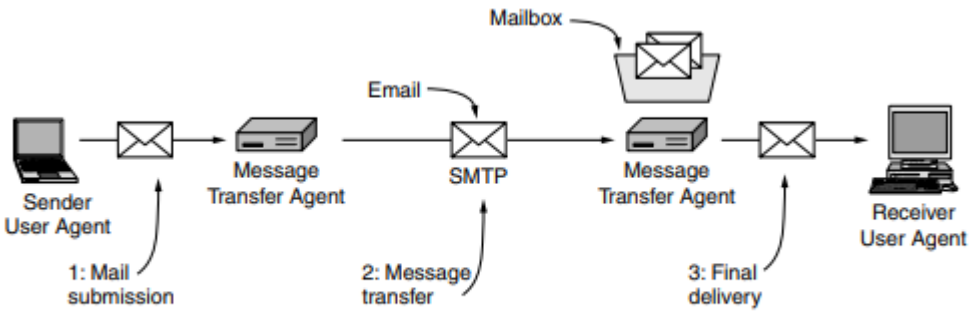


Figure 7-7. Architecture of the email system.

SMTP (Simple Mail Transfer Protocol)

- SMTP handles message transfer between mail servers.
- Defined initially in **RFC 821** and updated to **RFC 5321**.
- Provides delivery status and error reporting

Mailboxes and Message Access

- Email is stored in **mailboxes** located on mail servers.
- User agents access mailbox content by sending commands to the server.
- Retrieval of stored mail is considered the **final step in message delivery**.

Multiple user agents (phone, laptop, webmail) can access the **same mailbox**, allowing flexible usage.

Email Message Format

Email messages follow a standard structure.

- Originally defined by **RFC 822**
- Updated to **RFC 5322**
- Extended using **MIME** (Multipurpose Internet Mail Extensions) for:
    - Images
    - Audio/video
    - Non-English text
    - Attachments

A key idea in the message format is the distinction between the envelope and its contents. The envelope encapsulates the message. It contains all the information needed for transporting the message, such as the destination address, priority, and security level, all of which are distinct from the message itself. The message transport agents use the envelope for routing, just as the post office does.

The message inside the envelope consists of two separate parts: the header and the body. The header contains control information for the user agents. The body is entirely for the human recipient.

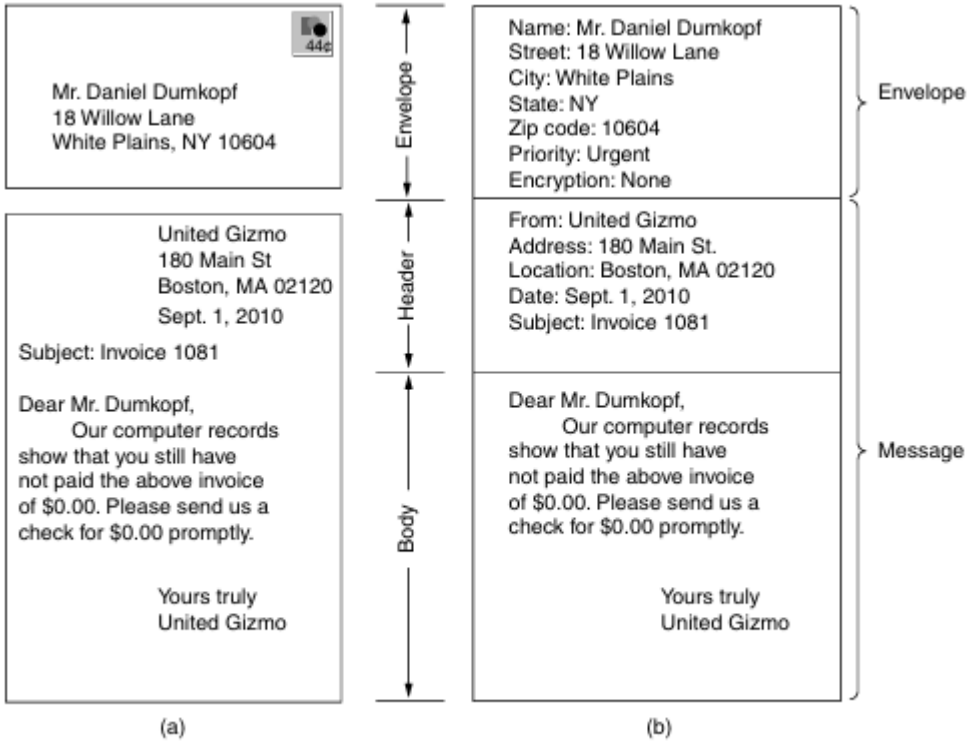


Figure 7-8. Envelopes and messages. (a) Paper mail. (b) Electronic mail.

The User Agent

- **A user agent is a program (sometimes called an email reader) that accepts a variety of commands for composing, receiving, and replying to messages, as well as for manipulating mailboxes.**
- There are many popular user agents, including Google Gmail, Microsoft Outlook, Mozilla Thunderbird, and Apple Mail.
- Most user agents have a menu- or icon driven graphical interface that requires a mouse, or a touch interface on smaller mobile devices.
- The typical elements of a user agent interface are shown in Fig. 7-9. Your mail reader is likely to be much flashier, but probably has equivalent functions.
- When a user agent is started, it will usually present a summary of the messages in the user's mailbox. Often, the summary will have one line for each message in some sorted order. It highlights key fields of the message that are extracted from the message envelope or header.
- The lines use the from, Subject, and Received fields, in that order, to display who sent the message, what it is about, and when it was received. All the information is formatted in a user-friendly way rather than displaying the literal contents of the message fields, but it is based on the message fields. Thus, people who fail to include a Subject field often discover that responses to their emails tend not to get the highest priority.

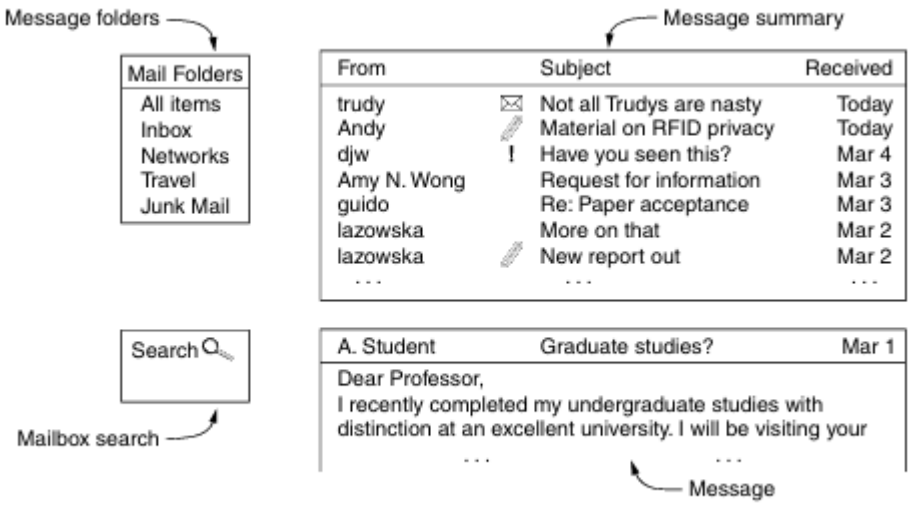


Figure 7-9. Typical elements of the user agent interface.

- User agents must also be able to display incoming messages as needed so that people can read their email. After a message has been read, the user can decide what to do with it. This is called **message disposition**. Options include deleting the message, sending a reply, forwarding the message to another user, and keeping the message for later reference. Most user agents can manage one mailbox for incoming mail with multiple folders for saved mail.

Message Format

RFC5322—The Internet Message Format

- RFC 5322 is a standard that defines the format of internet messages, such as email messages. It specifies the structure and content of email messages, including the headers, body, and attachments.
- The standard is maintained by the Internet Engineering Task Force (IETF) and is an important reference for anyone working with email or other internet messages. It is also known as the Internet Message Format Standard.
- The principal header fields related to message transport are listed in Fig. 7-10.

Header	Meaning
To:	Email address(es) of primary recipient(s)
Cc:	Email address(es) of secondary recipient(s)
Bcc:	Email address(es) for blind carbon copies
From:	Person or people who created the message
Sender:	Email address of the actual sender
Received:	Line added by each transfer agent along the route
Return-Path:	Can be used to identify a path back to the sender

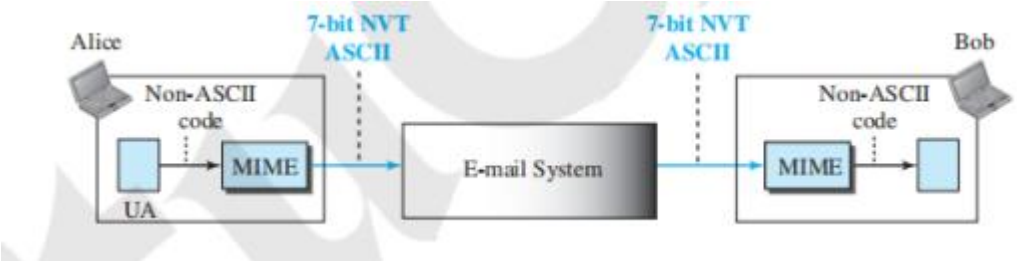
Figure 7-10. RFC 5322 header fields related to message transport.

- The To: field gives the DNS address of the primary recipient. Having multiple recipients is also allowed. The Cc: field gives the addresses of any secondary recipients. In terms of delivery, there is no distinction between the primary and secondary recipients. It is entirely a psychological difference that may be important to the people involved but is not important to the mail system. The term Cc: (Car bon copy) is a bit dated, since computers do not use carbon paper, but it is well established. The Bcc: (Blind carbon copy) field is like the Cc: field, except that this line is deleted from all the copies sent to the primary and secondary recipients. This feature allows people to send copies to third parties without the primary and secondary recipients knowing this.
- The next two fields, From: and Sender: tell who wrote and sent the message, respectively. These need not be the same. For example, a business executive may write a message, but her assistant may be the one who actually transmits it. In this case, the executive would be listed in the From: field and the assistant in the Sender: field. The From: field is required, but the Sender: field may be omit ted if it is the same as the From: field. These fields are needed in case the message is undeliverable and must be returned to the sender.

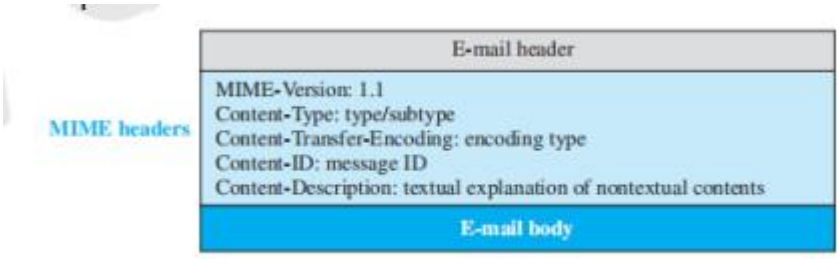
- A line containing Received: is added by each message transfer agent along the way. The line contains the agent's identity, the date and time the message was received, and other information that can be used for debugging the routing system. The Return-Path: field is added by the final message transfer agent and was intended to tell how to get back to the sender. In theory, this information can be gathered from all the Received: headers (except for the name of the sender's mail box), but it is rarely filled in as such and typically just contains the sender's ad dress.

MIME—The Multipurpose Internet Mail Extensions

- Email has a simple structure, but it is limited to sending messages in 7-bit ASCII format, which restricts its use for non-English languages and prevents sending binary files, video, or audio data.
- Multipurpose Internet Mail Extensions (MIME) is a supplementary protocol that allows non-ASCII data to be sent through e-mail. MIME transforms non-ASCII data at the sender site to NVT ASCII data and delivers it to the client MTA to be sent through the Internet. The message at the receiving site is transformed back to the original data. MIME is a set of software functions that transforms non-ASCII data to ASCII data and vice versa.



- **MIME Headers: -** MIME defines five headers, which can be added to the original e-mail header section to define the transformation parameters:



- **MIME-Version** -This header defines the version of MIME used. The current version is 1.1. Content-Type -This header defines the type of data used in the body of the message. The content type and the content subtype are separated by a slash. Depending on the subtype, the header may contain other parameters. MIME allows seven different types of data, listed in Table

Type	*Subtype*	*Description*
Text	Plain	Unformatted
	HTML	HTML format (see Appendix C)
Multipart	Mixed	Body contains ordered parts of different data types
	Parallel	Same as above, but no order
	Digest	Similar to Mixed, but the default is message/RFC822
	Alternative	Parts are different versions of the same message
Message	RFC822	Body is an encapsulated message
	Partial	Body is a fragment of a bigger message
	External-Body	Body is a reference to another message
Image	JPEG	Image is in JPEG format
	GIF	Image is in GIF format
Video	MPEG	Video is in MPEG format
Audio	Basic	Single channel encoding of voice at 8 KHz
Application	PostScript	Adobe PostScript
	Octet-stream	General binary data (eight-bit bytes)

Content-Transfer-Encoding This header defines the method used to encode the messages into 0s and 1s for transport. The five types of encoding methods are listed in Table

Type	*Description*
7-bit	NVT ASCII characters with each line less than 1000 characters
8-bit	Non-ASCII characters with each line less than 1000 characters
Binary	Non-ASCII characters with unlimited-length lines
Base64	6-bit blocks of data encoded into 8-bit ASCII characters
Quoted-printable	Non-ASCII characters encoded as an equal sign plus an ASCII code

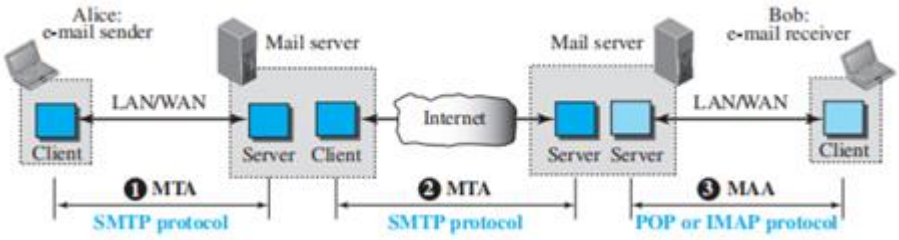
- **Content-ID** - This header uniquely identifies the whole message in a multiple message environment.
- **Content-Description** -This header defines whether the body is image, audio, or video

Message Transfer

- The simplest way to move messages is to establish a transport connection from the source machine to the destination machine and then just transfer the message. This is how SMTP originally worked.

SMTP (Simple Mail Transfer Protocol)

- Message Transfer Agent: SMTP an e-mail is an application that needs three uses of client-server paradigms to accomplish its task. It is important that we distinguish these three when we are dealing with e-mail. Figure shows these three client-server applications.
- We refer to the first and the second as Message Transfer Agents (MTAs), the third as Message Access Agent (MAA).
- **The formal protocol that defines the MTA client and server in the Internet is called Simple Mail Transfer Protocol (SMTP)**



- **Commands and Responses:** - SMTP uses commands and responses to transfer messages between an MTA client and an MTA server. The command is from an MTA client to an MTA server; the response is from an MTA server to the MTA client. Each command or reply is terminated by a two character (carriage return and line feed) end-of-line token.
- **Mail Transfer Phases**: - The process of transferring a mail message occurs in three phases: connection establishment, mail transfer, and connection termination.
- **Connection Establishment**: -After a client has made a TCP connection to the well-known port 25, the SMTP server starts the connection phase. This phase involves the following three steps:

  1. The server sends code 220 (service ready) to tell the client that it is ready to receive mail. If the server is not ready, it sends code 421 (service not available).
  2. The client sends the HELO message to identify itself, using its domain name address. This step is necessary to inform the server of the domain name of the client.
  3. The server responds with code 250 (request command completed) or some other code depending on the situation.

- **Message Transfer**: - After connection has been established between the SMTP client and server, a single message between a sender and one or more recipients can be exchanged. This phase involves eight steps. Steps 3 and 4 are repeated if there is more than one recipient.

  1. The client sends the MAIL FROM message to introduce the sender of the message. It includes the mail address of the sender (mailbox and the domain name). This step is needed to give the server the return mail address for returning errors and reporting messages.
  2. The server responds with code 250 or some other appropriate code.
  3. The client sends the RCPT TO (recipient) message, which includes the mail address of the recipient.
  4. The server responds with code 250 or some other appropriate code.
  5. The client sends the DATA message to initialize the message transfer.

6. The server responds with code 354 (start mail input) or some other appropriate message.

7. The client sends the contents of the message in consecutive lines. Each line is terminated by a two-character end-of-line token (carriage return and line feed). The message is terminated by a line containing just one period.

8. The server responds with code 250 (OK) or some other appropriate code.

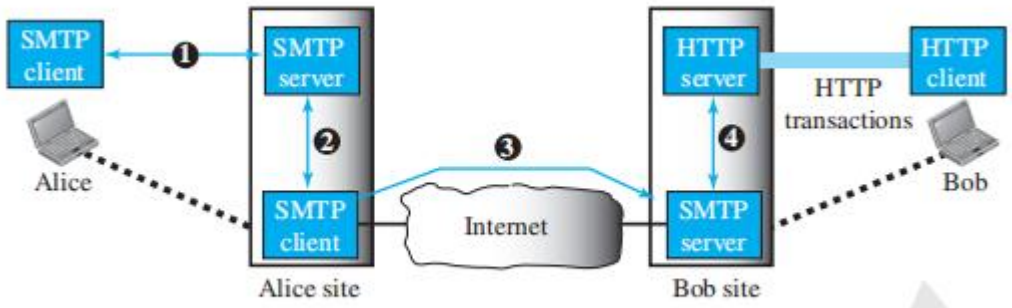
- **Connection Termination**: - After the message is transferred successfully, the client terminates the connection. This phase involves two steps.
  1. The client sends the QUIT command.
  2. The server responds with code 221 or some other appropriate code.

IMAP—The Internet Message Access Protocol

- IMAP4 Another mail access protocol is Internet Mail Access Protocol, version 4 (IMAP4). IMAP4 is similar to POP3, but it has more features; IMAP4 is more powerful and more complex.
- POP3 is deficient in several ways. It does not allow the user to organize her mail on the server; the user cannot have different folders on the server. In addition, POP3 does not allow the user to partially check the contents of the mail before downloading. IMAP4 provides the following extra functions:
  - A user can check the e-mail header prior to downloading.
  - A user can search the contents of the e-mail for a specific string of characters prior to downloading.
  - A user can partially download e-mail. This is especially useful if bandwidth is limited and the e-mail contains multimedia with high bandwidth requirements.
  - A user can create, delete, or rename mailboxes on the mail server.
  - A user can create a hierarchy of mailboxes in a folder for e-mail storage.

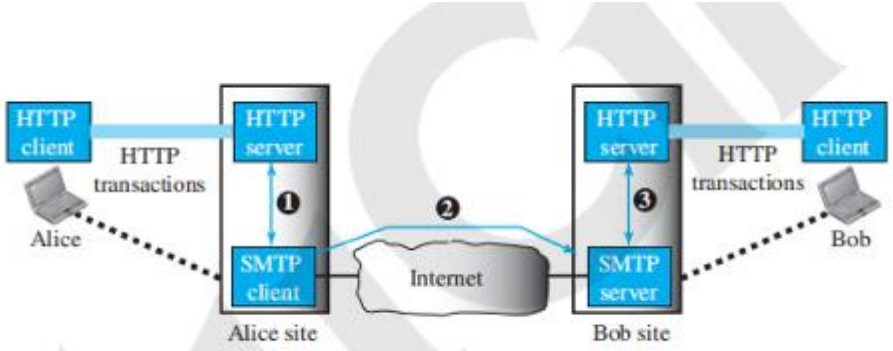
Web-Based Mail

- E-mail is such a common application that some websites today provide this service to anyone who accesses the site. Three common sites are Hotmail, Yahoo, and Google mail. The idea is very simple. Figure shows two cases:



Case 1: Only receiver uses HTTP

- **Case I: -** In this scenario, Alice uses a traditional mail server to send an email to Bob, who has an account on a web-based server. The email is transferred from Alice's browser to her mail server via SMTP, and from the sending mail server to the receiving mail server through SMTP as well. However, when the message reaches Bob's web server, it is transferred to Bob's browser via HTTP, not POP3 or IMAP4. Bob requests his emails through HTTP by logging into the website (e.g., Hotmail). Once logged in, the web server sends a list of emails in HTML format, and Bob can browse and retrieve his emails using more HTTP transactions.



Case 2: Both sender and receiver use HTTP

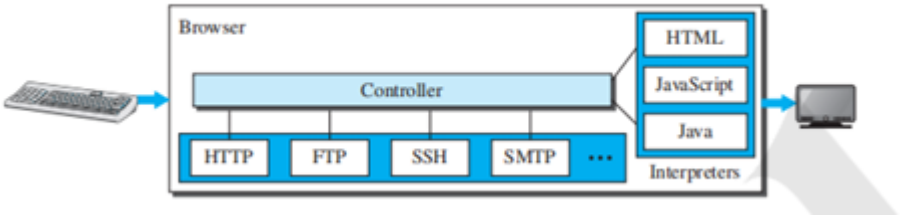
- **Case II: -** In this case, both Alice and Bob use web servers, but not necessarily the same one. Alice sends her email to her web server using HTTP, with Bob's mailbox address as the URL. The Alice server then forwards the message to Bob's server using SMTP. Bob receives the email through HTTP transactions, but the transfer between servers still occurs via SMTP.

The World Wide Web

- The idea of the Web was first proposed by Tim Berners-Lee in 1989 at CERN†, the European Organization for Nuclear Research, to allow several researchers at different locations throughout Europe to access each other's' researches.
- The commercial Web started in the early 1990s.
- Today, the Web is a global information space, where documents (called web pages) are spread across the world and connected through links.
- The Web's growth and popularity are tied to two key ideas: "distributed" and "linked."
    1. Distributed: Any web server worldwide can add new pages, which helps expand the Web without overwhelming a few servers.
    2. Linked: Web pages can reference other pages on different servers, creating a network of interconnected information.
- This linking is done through hypertext, a concept from before the Internet. It allows a document to automatically access another linked document. The Web made this possible electronically by letting users click a link to retrieve a connected document.
- The term hypertext evolved to hypermedia, meaning web pages can include not only text but also images, audio, and video.
- The Web's purpose has grown beyond sharing documents; it now supports online shopping, gaming, and on-demand access to radio and TV programs.

Architecture

- 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 sites. Each site holds one or more web pages. A web page can be simple or composite. A simple web page has no links to other web pages; a composite web page has one or more links to other web pages. Each web page is a file with a name and address.
- **Web Client (Browser): -** A variety of vendors offer commercial browsers that interpret and display a web page, and all of them use nearly the same architecture. Each browser usually consists of three parts: a controller, client protocols, 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 protocol can be one of the protocols described later, such as HTTP or FTP.
- The interpreter can be HTML, Java, or JavaScript, depending on the type of document.
- Some commercial browsers include Internet Explorer, Netscape Navigator, and Firefox.
- The web page is stored at the server. Each time a request arrives, the corresponding document is sent to the client.
- To improve efficiency, servers normally store requested files in a cache in memory; memory is faster to access than a disk.
- A server can also become more efficient through multithreading or multiprocessing. In this case, a server can answer more than one request at a time. Some popular web servers include Apache and Microsoft Internet Information Server.

Uniform Resource Locator (URL)

- A web page, as a file, needs to have a unique identifier to distinguish it from other web pages. To define a web page, four main parts are needed:
    1. Protocol: This is like choosing the "vehicle" or method to reach the web page. Commonly, this is HTTP (Hypertext Transfer Protocol), which lets the browser access and display web pages. However, other methods, like FTP (File Transfer Protocol), can also be used for different types of access.
    2. Host: The host tells us where the web page is stored. It can be written as an IP address (e.g., 64.23.56.17) or as a domain name, like "example.com," which makes it easier to identify servers.
    3. Port: The port number is a specific “entry point” on the server. For example, port 80 is usually used for HTTP, while HTTPS uses port 443. Most of the time, the port is standard and doesn’t need to be written, but if a different port is used, it can be specified in the address.
    4. Path: The path gives the exact location of the web page file within the server. It shows the series of folders and the file name, like “/folder1/folder2/filename.” This tells the server exactly where to look to find and deliver the web page.

Web Documents

- The documents in the WWW can be grouped into three broad categories: static, dynamic

Static Documents

- Static documents are fixed-content documents that are created and stored in a server. When a client accesses the document, a copy of the document is sent. The user can then use a browser to see the document. Static documents are prepared using one of several languages: Hypertext Mark-up Language (HTML), Extensible Mark-up Language (XML), Extensible Style Language (XSL), and Extensible Hypertext Mark-up Language (XHTML).

Dynamic Documents

- 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 or a script that creates the dynamic document. The server returns the result of the program or script 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 may vary from one request to another. A very simple example of a dynamic document is the retrieval of the time and date from a server.

HTML—The Hypertext MarkupLanguage

- lows users to produce Web pages that include text, graphics, video, pointers to other Web pages, and more. HTML is a mark-up language, or language for describing how documents are to be formatted. The term ‘‘mark-up’’ comes from the old days when copyeditors actually marked up documents to tell the printer— in those days, a human being—which fonts to use, and so on. Markup languages thus contain explicit commands for formatting. For example, in HTML, **means start boldface mode, and** means leave boldface mode. LaTeX and TeX are other examples of markup languages that are well known to most academic authors.
- HTML provides various mechanisms for making lists, including nested lists Unordered lists, like the ones in Fig. 7-23 are started with <ul>, with <li> used to mark the start of items. There is also an <ol> tag to starts an ordered list.

```
<html>
<head> <title> AMALGAMATED WIDGET, INC. </title> </head>
<body> <h1> Welcome to AWI's Home Page </h1>
<img src="http://www.widget.com/images/logo.gif" ALT="AWI Logo"> <br>
We are so happy that you have chosen to visit <b> Amalgamated Widget's</b>
home page. We hope <i> you </i> will find all the information you need here.
<p>Below we have links to information about our many fine products.
You can order electronically (by WWW), by telephone, or by email. </p>
<hr>
<h2> Product information </h2>
<ul>
   <li> <a href="http://widget.com/products/big"> Big widgets </a> </li>
   <li> <a href="http://widget.com/products/little"> Little widgets </a> </li>
</ul>
<h2> Contact information </h2>
<ul>
   <li> By telephone: 1-800-WIDGETS </li>
   <li> By email: info@amalgamated-widget.com </li>
</ul>
</body>
</html>
```

(a)

(b)

Figure 7-23. (a) The HTML for a sample Web page. (b) The formatted page.

AJAX—Asynchronous JavaScript and XML

- Compelling Web applications need responsive user interfaces and seamless access to data stored on remote Web servers. Scripting on the client (e.g., with JavaScript) and the server (e.g., with PHP) are basic technologies that provide pieces of the solution. These technologies are commonly used with several other key technologies in a

combination called AJAX (Asynchronous JavaScript and Xml). Many full-featured Web applications, such as Google's Gmail, Maps, and Docs, are written with AJAX.

- AJAX is somewhat confusing because it is not a language. It is a set of technologies that work together to enable Web applications that are every bit as responsive and powerful as traditional desktop applications. The technologies are:

    1. HTMLandCSS to present information as pages.

    2. DOM (Document Object Model) to change parts of pages while they are viewed.

    3. XML (eXtensible Markup Language) to let programs exchange ap plication data with the server.

    4. An asynchronous way for programs to send and retrieve XML data.

    5. JavaScript as a language to bind all this functionality together.
- DOM (Document Object Model) is a representation of an HTML page that is accessible to programs. This representation is structured as a tree that reflects the structure of the HTML elements.
- This element is the parent of the body element, which is in turn parent to a form element. The form has two attributes that are drawn to the right-hand side, one for the form method (a POST) and one for the form action (the URL to request). This element has three children, reflecting the two para graph tags and one input tag that are contained within the form. At the bottom of the tree are leaves that contain either elements or literals, such as text strings.

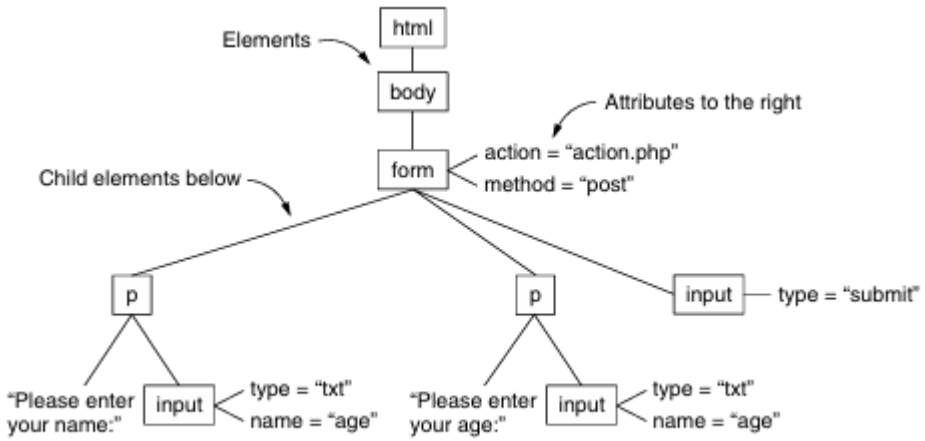


Figure 7-33. The DOM tree for the HTML in Fig. 7-30(a).

- **Hypertext Transfer Protocol (HTTP)**

- The Hyper Text Transfer Protocol (HTTP), the Web's application-layer protocol, is at the heart of the Web.

- HTTP is implemented in two programs: a client program and a server program. The client program and server program, executing on different end systems, talk to each other by exchanging HTTP messages. HTTP defines the structure of these messages and how the client and server exchange the messages.

- A Web page consists of objects. An object is simply a file like HTML file, a JPEG image, a Java applet, or a video clip—that is addressable by a single URL.

- Most Web pages consist of a base HTML file and several referenced objects.

- The base HTML file references the other objects in the page with the objects' URLs.

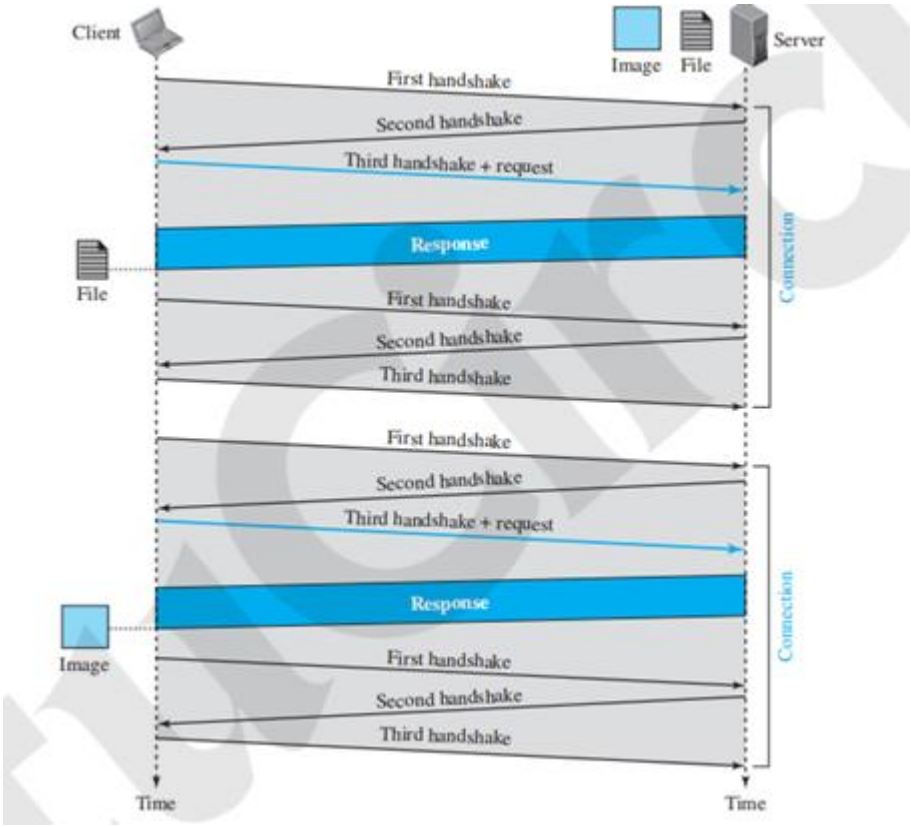
- HTTP defines how Web clients request Web pages from Web servers and how servers transfer Web pages to clients.

- When a user requests a Web page (for example, clicks on a hyperlink), the browser sends HTTP request messages for the objects in the page to the server. The server receives the requests and responds with HTTP response messages that contain the objects.

- HTTP uses TCP as its underlying transport protocol. The HTTP client first initiates a TCP connection with the server. Once the connection is established, the browser and the server processes access TCP through their socket interfaces Non-Persistent and Persistent Connections If Separate TCP connection is used for each request and response, then the connection is said to be non-persistent. If same TCP connection is used for series of related request and response, then the connection is said to be persistent.

Nonpersistent Connections

- In a nonpersistent connection, one TCP connection is made for each request/response. The following lists the steps in this strategy:
    1. The client opens a TCP connection and sends a request.
    2. The server sends the response and closes the connection.
    3. The client reads the data until it encounters an end-of-file marker; it then closes the connection.
- In this strategy, if a file contains links to N different pictures in different files (all located on the same server), the connection must be opened and closed N + 1 times. The nonpersistent strategy imposes high overhead on the server because the server needs N + 1 different buffers each time a connection is opened.
- Round-trip time (RTT) is the time it takes for a small packet to travel from client to server and then back to the client.
- The RTT includes packet-propagation delays, packet queuing delays in intermediate routers and switches, and packet-processing delays.
- When a user clicks on a hyperlink, the browser initiates a TCP connection between the browser and the Web server; this involves a "three-way handshake"—the client sends a small TCP segment to the server, the server acknowledges and responds with a small TCP segment, and, finally, the client acknowledges back to the server.



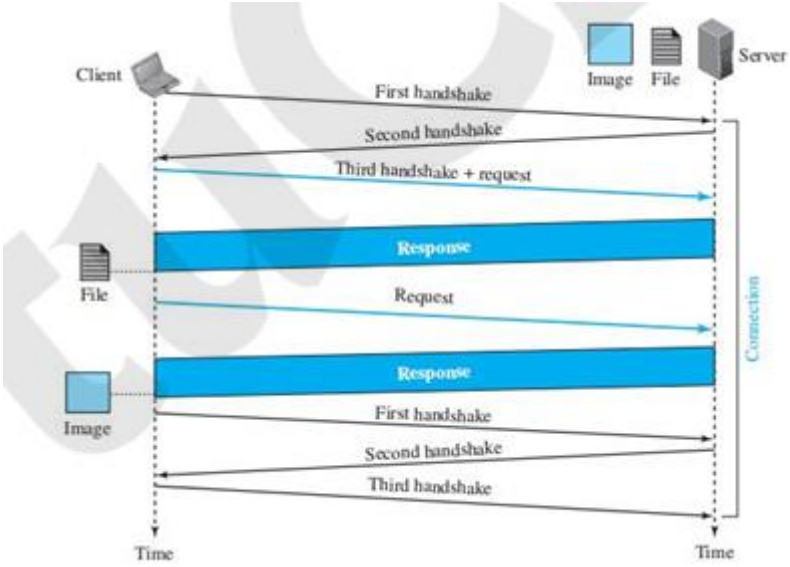
- The first two parts of the three-way handshake take one RTT.
- After completing the first two parts of the handshake, the client sends the HTTP request message combined with the third part of the three-way handshake (the acknowledgment) into the TCP connection.
- Once the request message arrives at the server, the server sends the HTML file into the TCP connection. This HTTP request/response eats up another RTT. Thus, roughly, the total response time is two RTTs plus the transmission time at the server of the HTML file.

Persistent Connections

- HTTP version 1.1 specifies a persistent connection by default. Non-persistent connections have some shortcomings.

  1. A brand-new connection must be established and maintained for each requested object. This can place a significant burden on the Web server, which may be serving requests from hundreds of different clients simultaneously.
  2. Each object suffers a delivery delay of two RTTs— one RTT to establish the TCP connection and one RTT to request and receive an object.

- With persistent connections, the server leaves the TCP connection open after sending a response. Subsequent requests and responses between the same client and server can be sent over the same connection. In particular, an entire Web page can be sent over a single persistent TCP connection. Moreover, multiple Web pages residing on the same server can be sent from the server to the same client over a single persistent TCP connection.
- Only one connection establishment and connection termination is used, but the request for the image is sent separately



- Request Message Method: There are five HTTP methods:
- ➢ GET: The GET method is used when the browser requests an object, with the requested object identified in the URL field.
- ➢ POST: With a POST message, the user is still requesting a Web page from the server, but the specific contents of the Web page depend on what the user entered into the form fields. If the value of the method field is POST, then the entity body contains what the user entered into the form fields.
- ➢ PUT: The PUT method is also used by applications that need to upload objects to Web servers.
- ➢ HEAD: Used to retrieve header information. It is used for debugging purpose.
- ➢ DELETE: The DELETE method allows a user, or an application, to delete an object on a Web server. URL: Specifies URL of the requested object Version: This field represents HTTP version, usually HTTP/1.1.

Web Caching

A Web cache—also called a proxy server—is a network entity that satisfies HTTP requests on the behalf of an origin Web server.

➢ The Web cache has its own disk storage and keeps copies of recently requested objects in this storage.

➢ A user's browser can be configured so that all of the user's HTTP requests are first directed to the Web cache.

Ex Suppose a browser is requesting the object http://www.someschool.edu/campus.gif. Here is what happens:

1. The browser establishes a TCP connection to the Web cache and sends an HTTP request for the object to the Web cache.

2. The Web cache checks to see if it has a copy of the object stored locally. If it does, the Web cache returns the object within an HTTP response message to the client browser

3. If the Web cache does not have the object, the Web cache opens a TCP connection to the origin server, that is, to www.someschool.edu. The Web cache then sends an HTTP request for the object into the cache-to-server TCP connection.

4. After receiving this request, the origin server sends the object within an HTTP response to the Web cache.

5. When the Web cache receives the object, it stores a copy in its local storage and sends a copy, within an HTTP response message, to the client browser (over the existing TCP connection between the client browser and the Web cache).

Web Search

- In 1998, Sergey Brin and Larry Page, then graduate students at Stanford, formed a start up called Google to build a better Web search engine. They were armed with the then-radical idea that a search algorithm that counted how many times each page was pointed to by other pages was a better measure of its importance than how many times it contained the key words being sought. For instance, many pages' link to the main Cisco page, which makes this page more important to a user searching for ''Cisco'' than a page out side of the company that happens to use the word ''Cisco'' many times.
- In one sense, search is simply another Web application, albeit one of the most mature Web applications because it has been under development since the early days of the Web. However, Web search has proved indispensable in everyday usage. Over one billion Web searches are estimated to be done each day. People looking for all manner of information use search as a starting point. For example, to find out where to buy Vegemite in Seattle, there is no obvious Web site to use as a starting point. But chances are that a search engine knows of a page with the desired information and can quickly direct you to the answer.
- To perform a Web search in the traditional manner, the user directs her browser to the URL of a Web search site. The major search sites include Google, Yahoo!, and Bing. Next, the user submits search terms using a form. This act causes the search engine to perform a query on its database for relevant pages or images, or whatever kind of resource is being searched for, and return the result as a dynamic page. The user can then follow links to the pages that have been found.
- Web search is an interesting topic for discussion because it has implications for the design and use of networks. First, there is the question of how Web search finds pages. The Web search engine must have a database of pages to run a query. Each HTML page may contain links to other pages, and everything interesting (or at least searchable) is linked somewhere. This means that it is theoretically possible to start with a handful of pages and find all other pages on the Web by doing a traversal of all pages and links. This process is called Web crawling. All Web search engines use Web crawlers.