UNIT II Application Layer

Principles

WEB And HTTP

STATIC WEB DOCUMENTS

 The basis of the Web is transferring Web pages from server to client. In the simplest form, Web pages are static. They are just files sitting on some server waiting to be retrived.

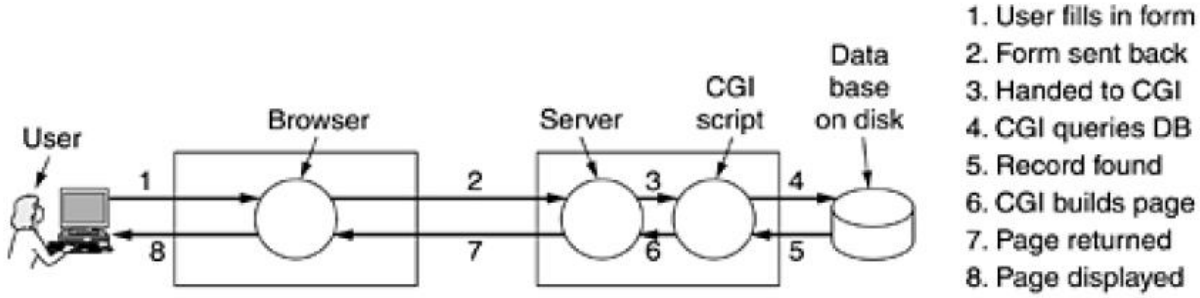
 In this context, even a video is a static web page because it is just a file.

 In this section we will look at static web page in details. In the next one, we will examine dynamic content.

DYNAMIC WEB DOCUMENTS

Dynamic web documents are created at both client and server sides.

SERVER-SIDE GENERATION



The server side generation involves the following steps:

 User fills in form.

 Form sent back

 Handed to CGI

 CGI queries database.

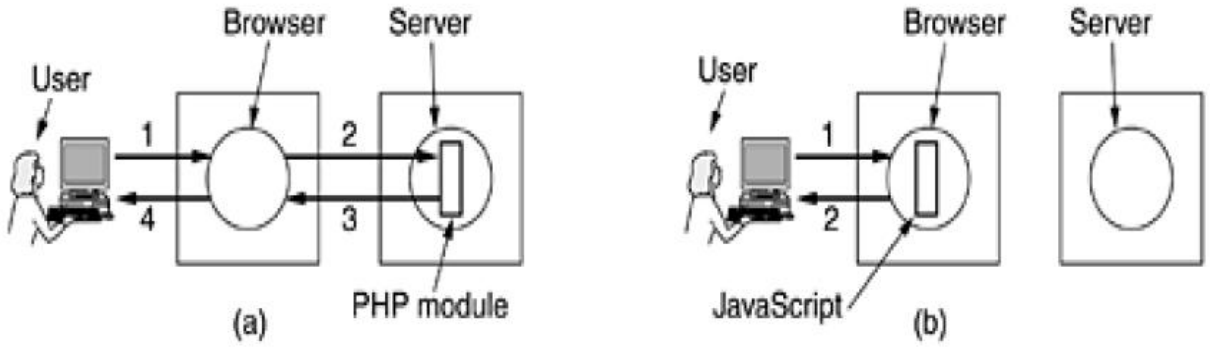
 Record found

 CGI builds page

 Page returned

 Page displayed

CLIENT-SIDE GENERATION



CGI, PHP, JSP, and ASP scripts solve the problem of handling forms and interactions with databases on the server. They can all accept incoming information from forms, look up information in one or more databases, and generate HTML pages with the results.
Usually the server side scripting is done with PHP and client side scripting is javascript. Complete web pages can be generated on the fly by various scripts on the server machine. Once they are received by the browser, they are treated as normal HTML pages and displayed.
Dynamic content generation is also possible on the client side. Web pages can be written in XML and then converted to HTML according to XSL file. Javascript programs can perform arbitrary computations.
Finally plugins and helper applications can be used to display content in a variety of formats.

Hypertext Transfer Protocol (HTTP)
The Hypertext Transfer Protocol (HTTP) is an application protocol for distributed, collaborative, hypermedia information systems. HTTP is the foundation of data communication for the World Wide Web.
HTTP functions as a request-response protocol in the client-server computing model. A web browser, for example, may be the client and an application running on a computer hosting a web site may be the server. The client submits an HTTP request message to the server. The server, which provides resources such as HTML files and other content, or performs other functions on behalf of the client, returns a response message to the client. The response contains completion status information about the request and may also contain requested content in its message body.
HTTP resources are identified and located on the network by Uniform Resource Identifiers (URIs)—or, more specifically, Uniform Resource Locators (URLs)—using the http or https URI schemes. URIs and hyperlinks in Hypertext Mark up Language (HTML) documents form webs of inter-linked hypertext documents. On

the Internet the World Wide Web was established in 1990 by English computer scientist and innovator Tim Berners-Lee.

DOMAIN NAME SYSTEM

This is primarily used for mapping host and e-mail destinations to IP addresses but can also be used other purposes. DNS is defined in RFCs 1034 and 1035.

Working:-

 To map a name onto an IP address, an application program calls a library procedure called Resolver, passing it the name as a parameter.

 The resolver sends a UDP packet to a local DNS server, which then looks up the name and returns the IP address to the resolver, which then returns it to the caller.

 Armed with the IP address, the program can then establish a TCP connection with the destination, or send it UDP packets.

1. The DNS name space.

2. Resource Records.

3. Name Servers.

1. THE DNS NAME SPACE:

The Internet is divided into several hundred top level domains, where each domain covers many hosts. Each domain is partitioned into sub domains, and these are further partitioned as so on. All these domains can be represented by a tree, in which the leaves represent domains that have no sub domains. A leaf domain may contain a single host, or it may represent a company and contains thousands of hosts. Each domain is named by the path upward from it to the root. The components are separated by periods (pronounced "dot")

Eg: Sun Microsystems Engg. Department = eng.sun.com.

The top domain comes in 2 flavours:-

 Generic: com(commercial), edu(educational instructions), mil(the U.S armed forces, government), int (certain international organizations), net(network providers), org (non profit organizations).

- **Country:** include 1 entry for every country. Domain names can be either absolute (ends with a period e.g. eng.sum.com) or relative (doesn't end with a period). Domain names are case sensitive and the component names can be up to 63 characters long and full path names must not exceed 255 characters.

Domain	Name	Time to live	Type	Class	Value

Figure 5-1. A portion of the Internet domain name space.

Insertions of a domain into the tree can be done in 2 days:-

• Under a generic domain (Eg: cs.yale.edu)
• Under the domain of their country (E.g: cs.yale.ct.us)

2. RESOURCE RECORDS:

Every domain can have a sent of resource records associated with it. For a single host, the most common resource record is just its IP address. When a resolver gives a domain name to DNS, it gets both the resource records associated with that name i.e., the real function of DNS is to map domain names into resource records.

A resource record is a 5-tuple and its format is as follows:

Domain _name : Tells the domain to which this record applies.
Time- to- live : Gives an identification of how stable the record is (High Stable = 86400 i.e. no. of seconds /day) (High Volatile = 1 min)
Type: Tells what kind of record this is.
Class: It is IN for the internet information and codes for non internet information
Value: This field can be a number a domain name or an ASCII string

Type	Meaning	Value
SOA	Start of Authority	Parameters for this zone
A	IP address of a host	32-Bit integer
MX	Mail exchange	Priority, domain willing to accept e-mail
NS	Name Server	Name of a server for this domain
CNAME	Canonical name	Domain name
PTR	Pointer	Alias for an IP address
HINFO	Host description	CPU and OS in ASCII
TXT	Text	Uninterpreted ASCII text

NAME SERVERS:

It contains the entire database and responds to all queries about it. DNS name space is divided up into non-overlapping zones, in which each zone contains some part of the tree and also contains name servers holding the authoritative information about that zone.

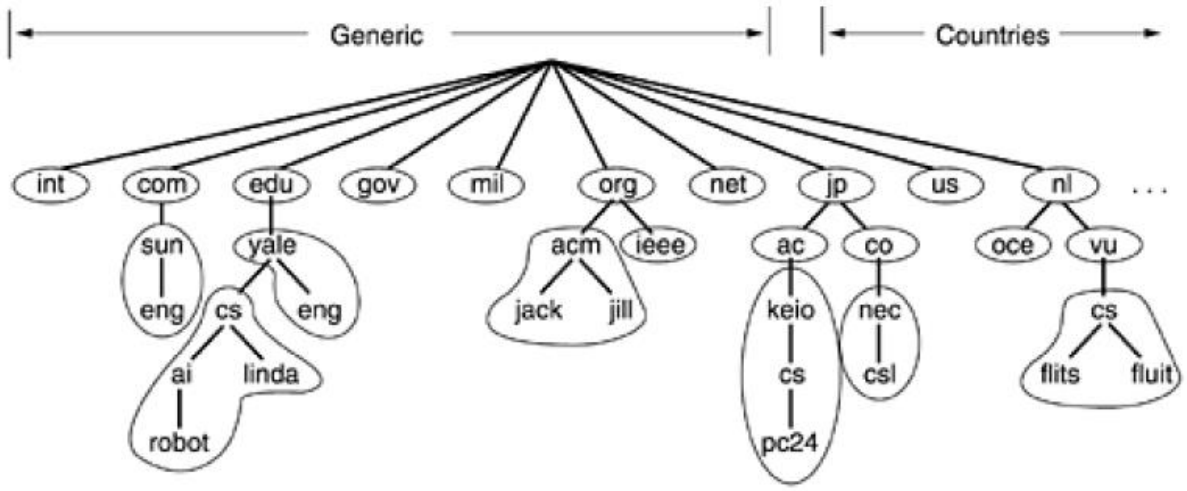


Figure 5-2. Part of the DNS name space showing the division into zones.

When a resolver has a query about a domain name, it passes the query to one of the local name servers:

1. If the domain being sought falls under the jurisdiction of name server, it returns the authoritative resource records (that comes from the authority that manages the record,and is always correct).

2. If the domain is remote and no information about the requested domain is available locally the name server sends a query message to the top level name server for the domain requested.

E.g.: A resolver of flits.cs.vle.nl wants to know the IP address of the host Linda.cs.yale.edu

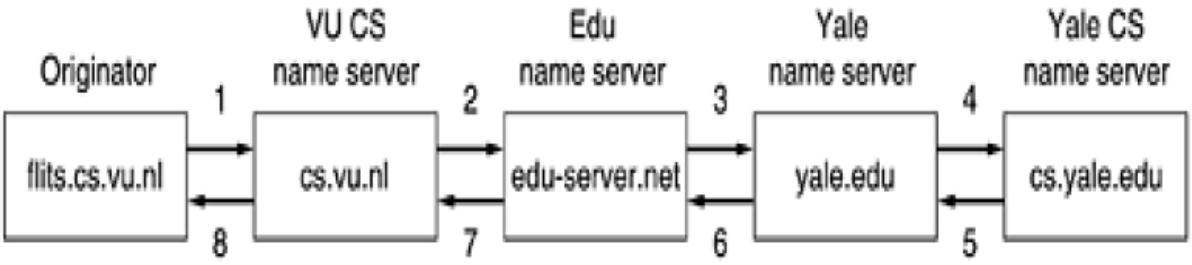


Figure 5-3. How a resolver looks up a remote name in eight steps.

Step 1: Resolver sends a query containing domain name sought the type and the class to local name server, cs.vu.nl.

Step 2: Suppose local name server knows nothing about it, it asks few others nearby name servers. If none of them know, it sends a UDP packet to the server for edu-server.net.
Step 3: This server knows nothing about Linda.cs.yale.edu or cs.yale.edu and so it forwards the request to the name server for yale.edu.
Step 4: This one forwards the request to cs.yale.edu which must have authoritative resource records.
Step 5 to 8: The resource record requested works its way back in steps 5-8 This query method is known as **Recursive Query**
3. When a query cannot be satisfied locally, the query fails but the name of the next server along the line to try is returned.

ELECTRONIC MAIL

1. ARCHITECTURE AND SERVICES:

E-mail systems consist of two subsystems. They are:-
(1). User Agents, which allow people to read and send e-mail
(2). Message Transfer Agents, which move messages from source to destination
E-mail systems support 5 basic functions:-
a. Composition
b. Transfer
c. Reporting
d. Displaying
e. Disposition

(a). Composition: It refers to the process of creating messages and answers. Any text editor is used for body of the message. While the system itself can provide assistance with addressing and numerous header fields attached to each message.
(b). Reporting: It has to do with telling the originator what happened to the message that is, whether it was delivered, rejected (or) lost.
(c). Transfer: It refers to moving messages from originator to the recipient.
(d). Displaying: Incoming messages are to be displayed so that people can read their email.
(e). Disposition: It concerns what the recipient dose with the message after receiving it. Possibilities include throwing it away before reading (or) after reading, saving it and so on.

Most systems allow users to create **mailboxes** to store incoming e-mail. Commands are needed to create and destroy mailboxes, inspect the contents of mailboxes, insert and delete messages from mailboxes, and so on.

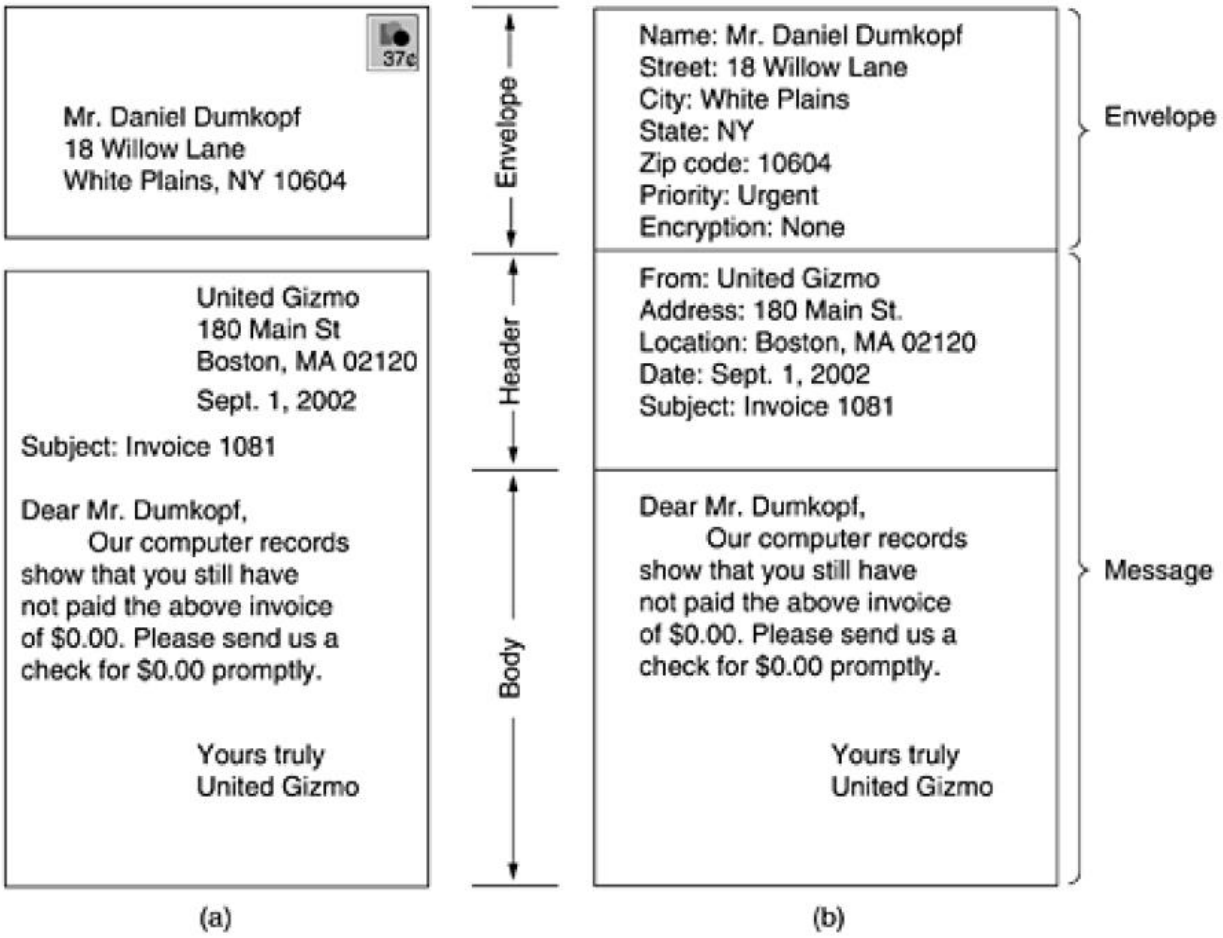


Figure 5-4: Envelopes and messages. (a) Paper mail. (b) Electronic mail.

(1) THE USER AGENT

A user agent is normally a program (sometimes called a mail reader) that accepts a variety of commands for composing, receiving, and replying to messages, as well as for manipulating mailboxes.

SENDING E-MAIL

To send an e-mail message, a user must provide the message, the destination address, and possibly some other parameters. The message can be produced with a free-standing text editor, a word processing program, or possibly with a specialized text editor built into the user agent. The destination address must be in a format that the user agent can deal with. Many user agents expect addresses of the form *user@dns-address*.

READING E-MAIL

When a user agent is started up, it looks at the user's mailbox for incoming e-mail before displaying anything on the screen. Then it may announce the number of messages in the mailbox or display a one-line summary of each one and wait for a command.

(2) MESSAGE FORMATS

RFC 822

Messages consist of a primitive envelope (described in RFC 821), some number of header fields, a blank line, and then the message body. Each header field (logically) consists of a single line of ASCII text containing the field name, a colon, and, for most fields, a value.

Header	Meaning
To:	E-mail address(es) of primary recipient(s)
Cc:	E-mail address(es) of secondary recipient(s)
Bcc:	E-mail address(es) for blind carbon copies
From:	Person or people who created the message
Sender:	E-mail 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 5-5: RFC 822 header fields related to message transport

MIME — The Multipurpose Internet Mail Extensions

RFC 822 specified the headers but left the content entirely up to the users. Nowadays, on the worldwide Internet, this approach is no longer adequate. The problems include sending and receiving

1. Messages in languages with accents (e.g., French and German).
2. Messages in non-Latin alphabets (e.g., Hebrew and Russian).
3. Messages in languages without alphabets (e.g., Chinese and Japanese).
4. Messages not containing text at all (e.g., audio or images).

A solution was proposed in RFC 1341 called **MIME** (**Multipurpose Internet Mail Extensions**)

The basic idea of MIME is to continue to use the RFC 822 format, but to add structure to the message body and define encoding rules for non-ASCII messages. By not deviating from RFC 822, MIME messages can be sent using the existing mail programs and protocols. All that has to be changed are the sending and receiving programs, which users can do for themselves.

Header	Meaning
MIME-Version:	Identifies the MIME version
Content-Description:	Human-readable string telling what is in the message
Content-Id:	Unique identifier
Content-Transfer-Encoding:	How the body is wrapped for transmission
Content-Type:	Type and format of the content

Figure 5-6: RFC 822 headers added by MIME

MESSAGE TRANSFER

The message transfer system is concerned with relaying messages from the originator to the recipient. The simplest way to do this is to establish a transport connection from the source machine to the destination machine and then just transfer the message.

SMTP—THE SIMPLE MAIL TRANSFER PROTOCOL

SMTP is a simple ASCII protocol. After establishing the TCP connection to port 25, the sending machine, operating as the client, waits for the receiving machine, operating as the server, to talk first. The server starts by sending a line of text giving its identity and telling whether it is prepared to receive mail. If it is not, the client releases the connection and tries again later.

Even though the SMTP protocol is completely well defined, a **few problems** can still arise.

One problem relates to message length. Some older implementations cannot handle messages exceeding 64 KB.

Another problem relates to timeouts. If the client and server have different timeouts, one of them may give up while the other is still busy, unexpectedly terminating the connection.

Finally, in rare situations, infinite mailstorms can be triggered.

For example, if host 1 holds mailing list *A* and host 2 holds mailing list *B* and each list contains an entry for the other one, then a message sent to either list could generate a never-ending amount of e-mail traffic unless somebody checks for it.

FINAL DELIVERY

With the advent of people who access the Internet by calling their ISP over a modem, it breaks down.

One solution is to have a message transfer agent on an ISP machine accept e-mail for its customers and store it in their mailboxes on an ISP machine. Since this agent can be on-line all the time, e-mail can be sent to it 24 hours a day.

Socket programming with TCP and UDP

The socket primitives are mainly used for TCP. These sockets were first released as part of the Berkeley UNIX 4.2BSD software distribution in 1983. They quickly became popular. The primitives are now widely used for Internet programming on many operating systems, especially UNIX -based systems, and there is a socket-style API for Windows called ‘‘**winsock**.’’

Primitive	Meaning
SOCKET	Create a new communication end point
BIND	Attach a local address to a socket
LISTEN	Announce willingness to accept connections; give queue size
ACCEPT	Block the caller until a connection attempt arrives
CONNECT	Actively attempt to establish a connection
SEND	Send some data over the connection
RECEIVE	Receive some data from the connection
CLOSE	Release the connection

The first four primitives in the list are executed in that order by servers.
The **SOCKET** primitive creates a new endpoint and allocates table space for it within the transport entity. The parameter includes the addressing format to be used, the type of service desired and the protocol. Newly created sockets do not have network addresses.

- The **BIND** primitive is used to connect the newly created sockets to an address. Once a server has bound an address to a socket, remote clients can connect to it.
- The **LISTEN** call, which allocates space to queue incoming calls for the case that several clients try to connect at the same time.
- The server executes an **ACCEPT** primitive to block waiting for an incoming connection.

Some of the client side primitives are. Here, too, a socket must first be created

- The **CONNECT** primitive blocks the caller and actively starts the connection process. When it completes, the client process is unblocked and the connection is established.
- Both sides can now use **SEND** and **RECEIVE** to transmit and receive data over the full-duplex connection.
- Connection release with sockets is symmetric. When both sides have executed a **CLOSE** primitive, the connection is released.