ELECTRONIC MAIL

- Most widely used application on the Internet.
- **For sending mails:**
  - Simple Mail Transfer Protocol (SMTP)
  - Multi-purpose Internet Mail Extension (MIME)
- **For receiving mails:**
  - Post office protocol version 3 (POP3)
  - Internet mail access protocol (IMAP).

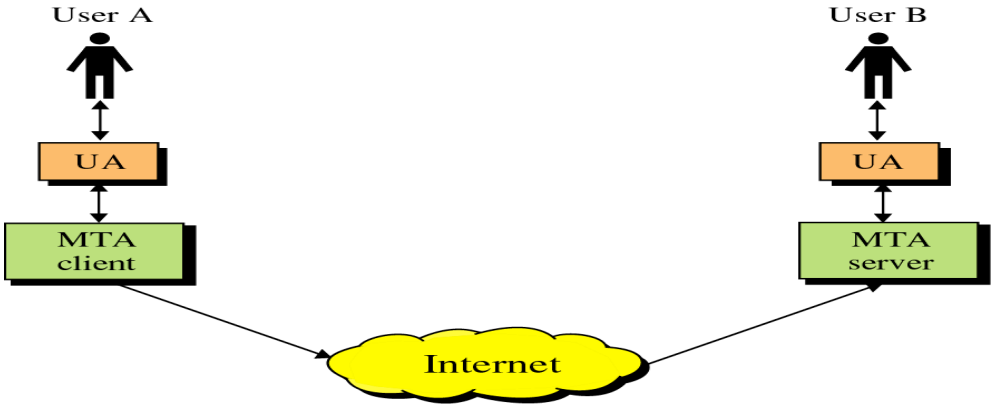
SIMPLE MAIL TRANSFER PROTOCOL (SMTP)

Simple Mail Transfer Protocol (**SMTP**) is an Internet standard for electronic mail (email) transmission. First defined by RFC 821 in 1982, it was last updated in 2008 with the Extended **SMTP** additions by RFC 5321-which is the protocol in widespread use today. **SMTP** by default uses TCP port 25.

- Transmits simple text messages only.
  - ✓ 7-bit ASCII format.
- Uses information written on envelope of mail.
  - ✓ Message header.

SMTP CLIENTS AND SERVERS HAVE TWO MAIN COMPONENTS:

- **User Agents** – Prepares the message, encloses it in an envelope. (ex. Thunderbird, Eudora)
- **Mail Transfer Agent** – Transfers the mail across the internet (ex. Sendmail, Exim)
- Analogous to the postal system in many ways



FORMAT OF AN E-MAIL:

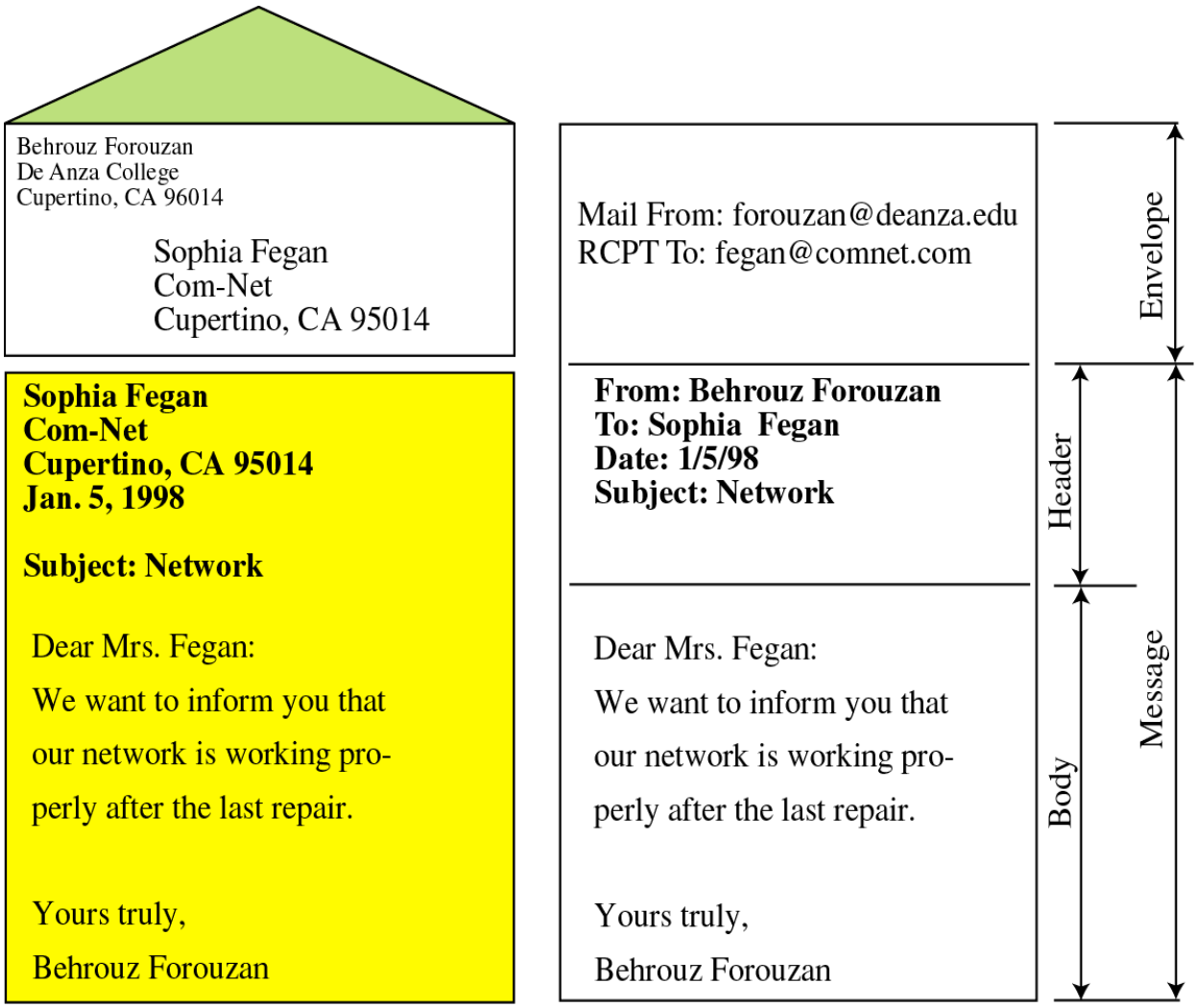
- Mail is a Text File.

Envelope: It contains with sender address, receiver address and other information.

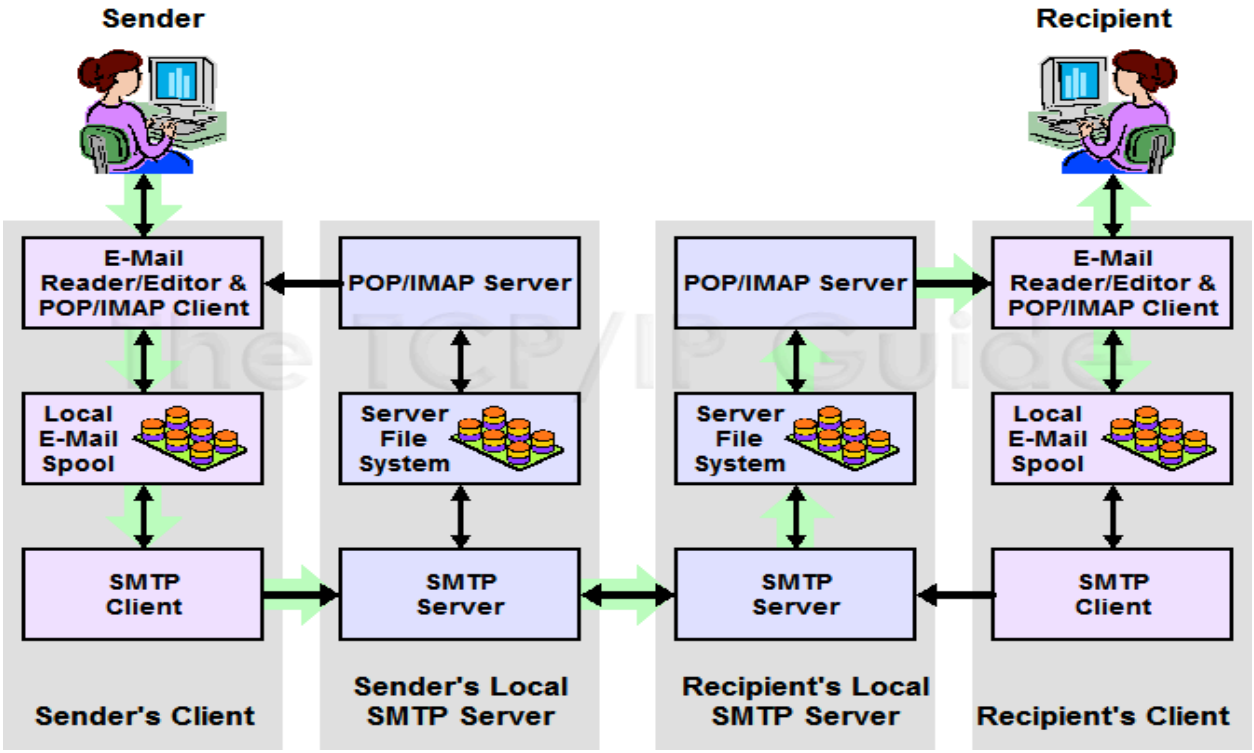
Message: It contains Mail Header and Mail Body.

Mail Header: It defines the sender, the receiver, the subject of the message and other information.

Mail Body: It contains the actual information in the message



SMTP COMMUNICATION MODEL



COMMANDS OF SMTP:

HELO : Request to initiate SMTP session

MAIL FROM: Sender's E-Mail address

RCPT TO : Receiver's E-Mail address

DATA : Body of message

QUIT : Terminates SMTP connection

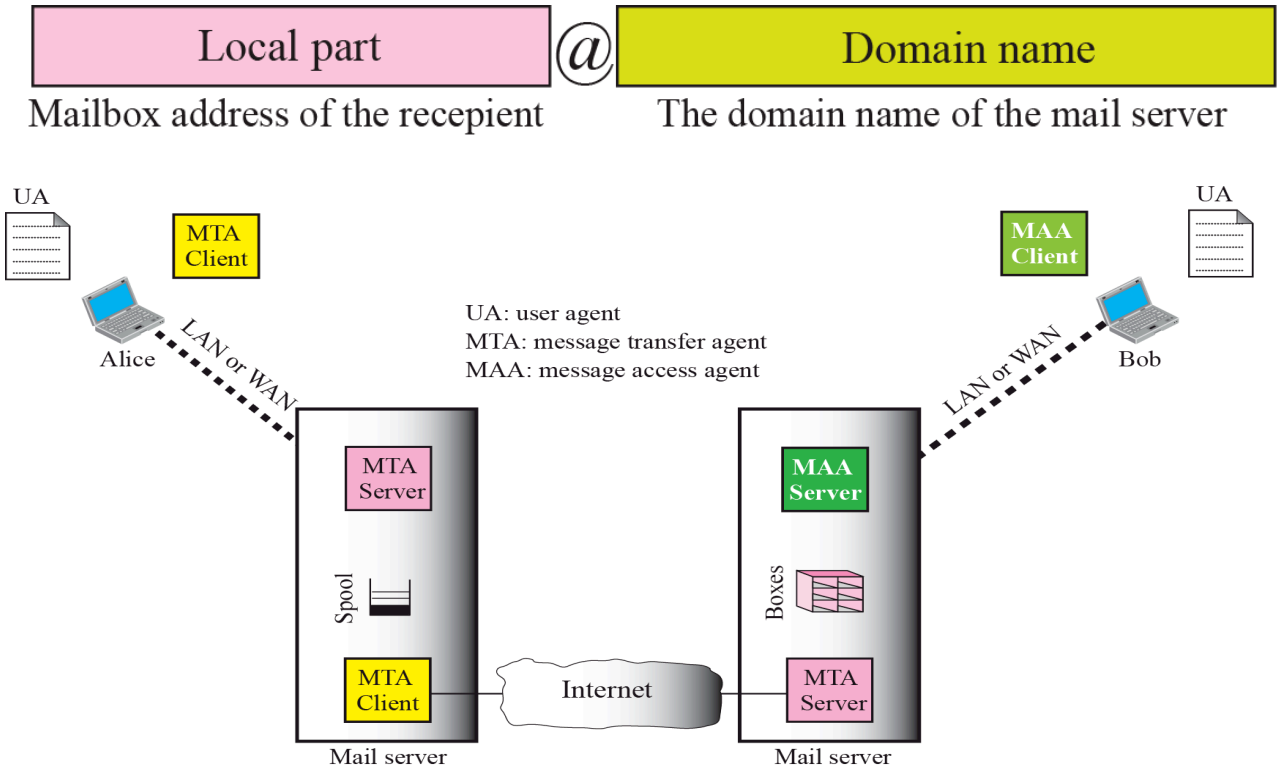
RSET : Aborts mail transaction

VRFY : Asks receiver to verify the validity of the mailbox

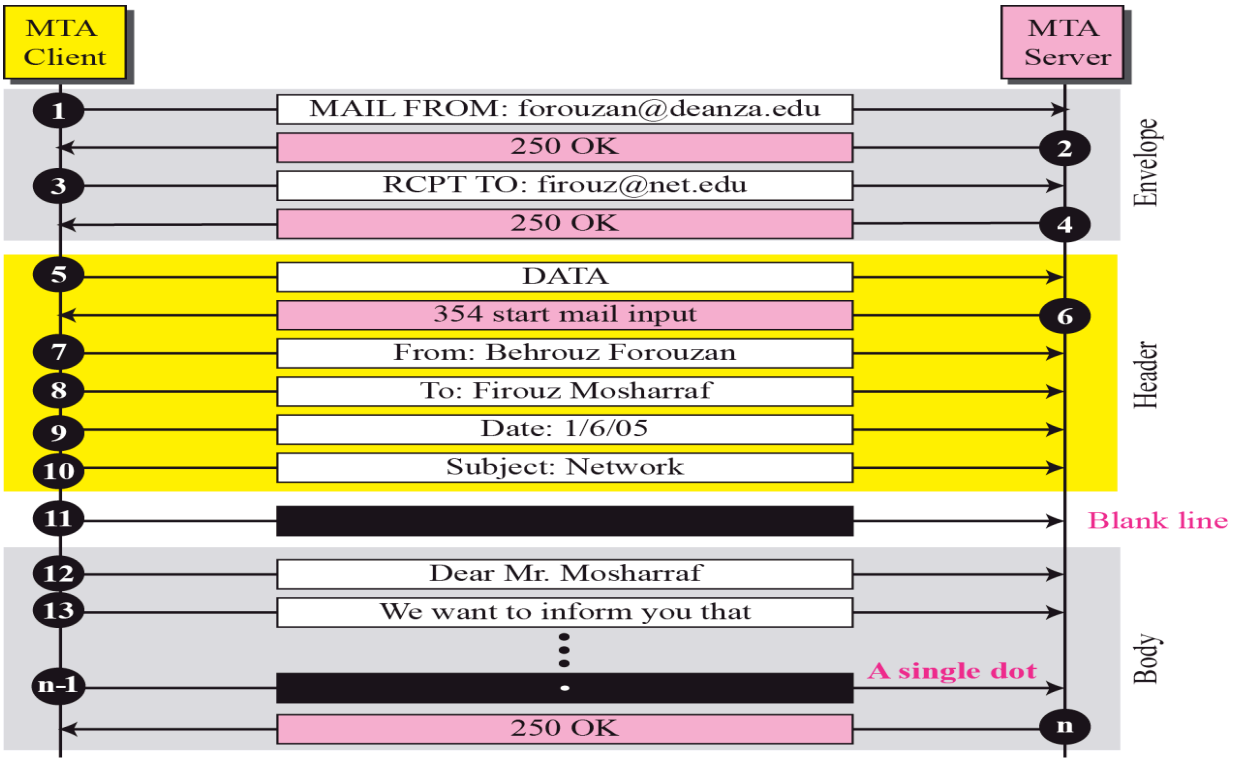
EXPN : Asks receiver to identify mailing list

HELP : Causes receiver to send help information

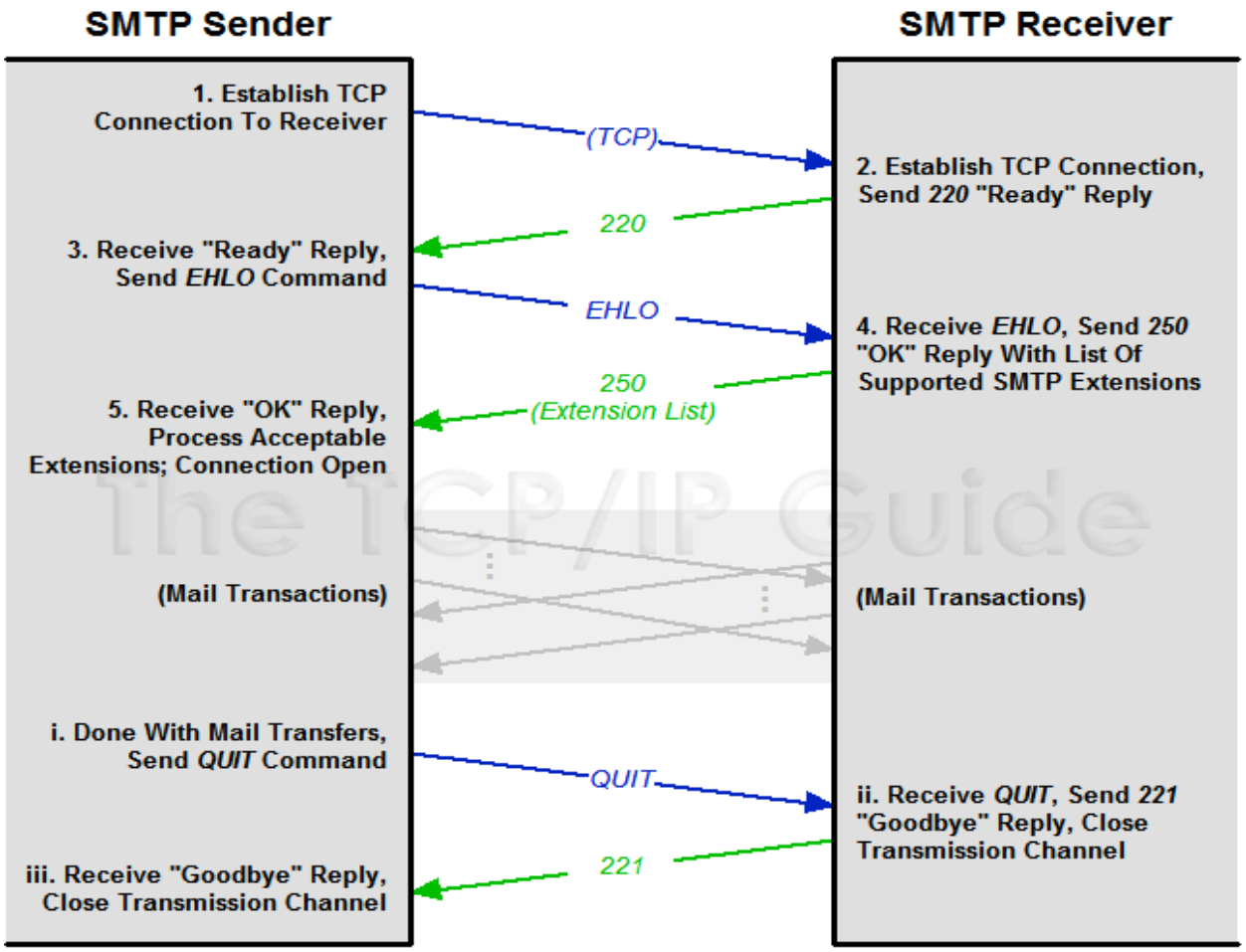
NOOP : Forces server to verify the communication with SMTP receiver



MESSAGE TRANSFER:



SESSION ESTABLISHMENT AND TERMINATION:



STATUS CODES:

The Server responds with a three digit code that may be followed by the text info.

- 2XX – The SMTP server has accepted the command and has completed the request.
- 3XX - Command is accepted and more information follows.
- 4XX - Try again later as there was a temporary failure with the command or the server.
- 5XX – The requested operation will never be completed due to permanent error.

ADVANTAGES

- Very Popular
- Supported on many platforms
- Low administration and implementation costs
- Simple addressing scheme

LIMITATIONS:

- Security matters for SMTP are worse.
- Its usefulness is limited by its simplicity.
- Transmission of executable files and binary files using SMTP is not possible without converting into text files. Use MIME to send mail in other format.
- It cannot transmit text data that contains national language characters. These national language characters use 8-bit codes with values of 128 decimal or more.
- It is limited to 7-bit ASCII characters only.
- SMTP servers may reject mail messages beyond some specific length.

SPECIAL FEATURES:

- **Mail Forwarding**: SMTP server may agree to accept e-mail for non-local mailbox and forward it to the appropriate destination.
- **Mail Gatewaying:** SMTP servers can be implemented as e-mail gateways which can translate TCP/IP email in a suitable form for some another e-mail system and vice-versa.
- Mail Relaying: SMTP includes the ability to relay mail from one server to another, as explained earlier, provided certain conditions are met.

- **Address Debugging:** VRFY command allows the client to ask the sender to verify address of recipient without sending mail to recipient.
- **Mailing List Expansion:** EXPN command allows to expand mailing list.

CONCLUSION:

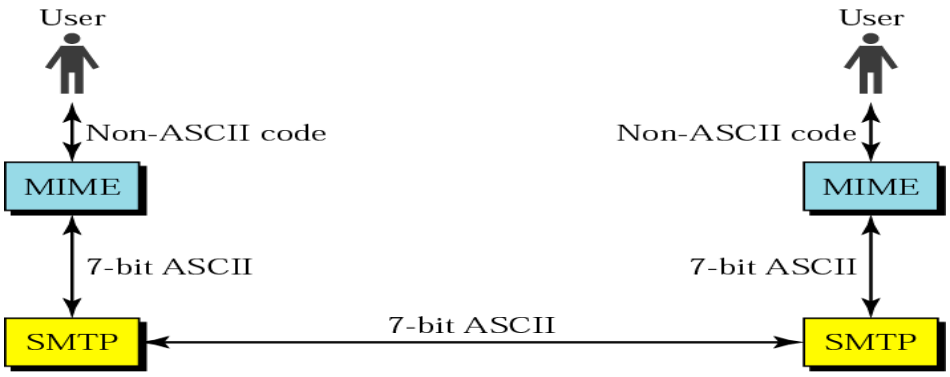
SMTP is one of the most widely used and implemented application. With the explosively growing reliance on electronic mail for commercial and personal services, there grows the demand of authentication and confidentiality. To complement the weak security feature of SMTP industry use PGP-SMIME-PEM. Still there is need of implementing the measures to eliminate spam and other security breaches.

MULTI-PURPOSE INTERNET MAIL EXTENSION (MIME)

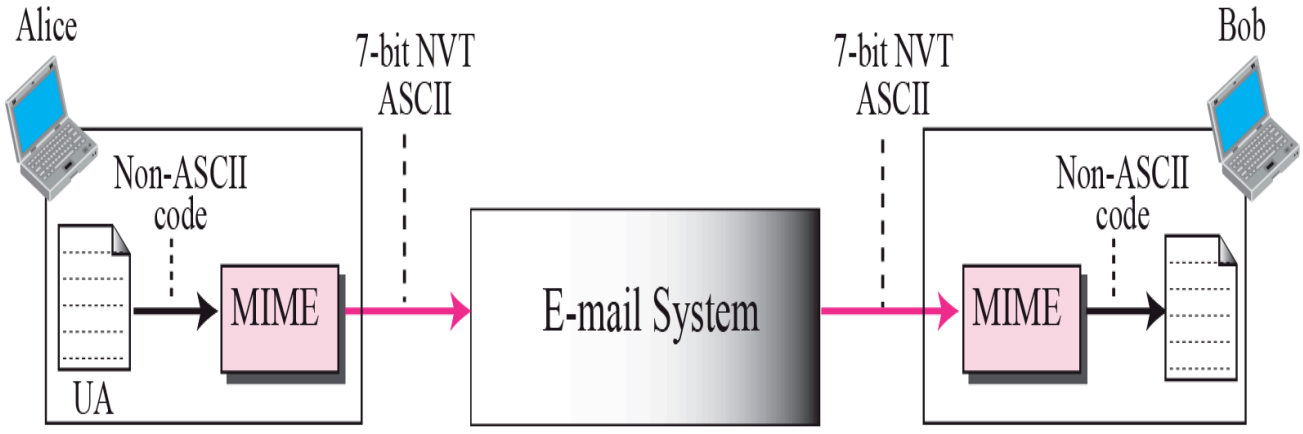
MIME (Multi-Purpose Internet Mail Extensions) is an extension of the original Internet e-mail **protocol** that lets people use the **protocol** to exchange different kinds of data files on the Internet: audio, video, images, application programs, and other kinds, as well as the ASCII text handled in the original **protocol.**

Multipurpose Internet Mail Extensions (**MIME**) is an Internet standard that extends the format of email to support:

- Text in character sets other than ASCII
- Non-text attachments: audio, video, images, application programs etc.
- Message bodies with multiple parts
- Header information in non-ASCII character sets



- A protocol for transmitting non-text information across the Internet. Basically, non-ASCII data is converted to ASCII for transmission and then converted back at the receiver.
- A specification for automatically sending objects other than text in email messages.
- MIME is usually associated with multimedia, such as images, audio recordings, and movies.
- Additional hardware and helper software are usually required.
- Common MIME-compliant mailers:
    - pine, metamail, Netscape messenger, MS Outlook



MIME header:

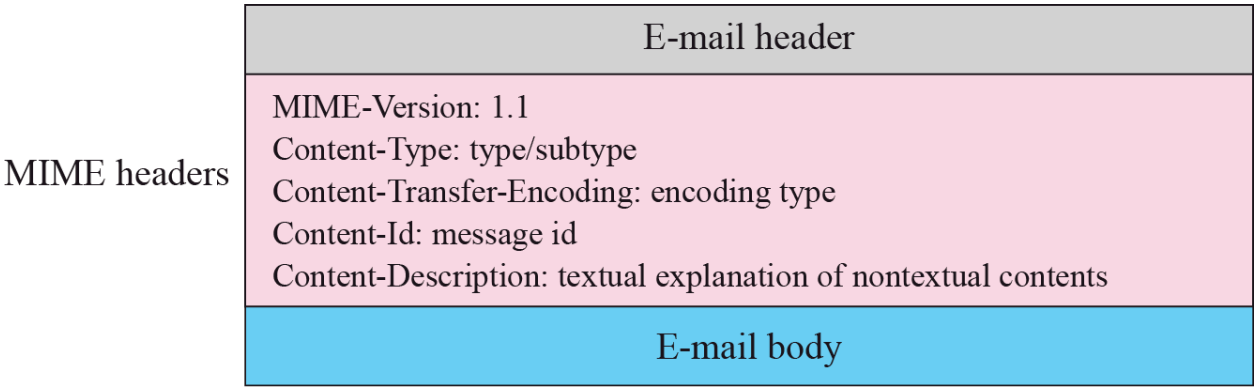
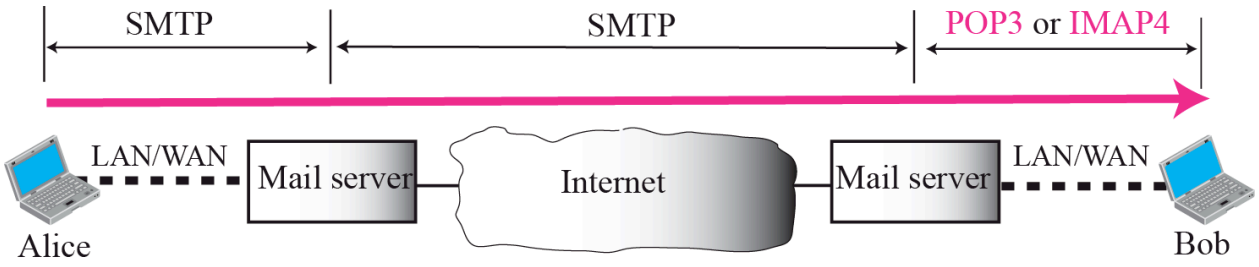


Table 23.3 *Data Types and Subtypes in MIME*

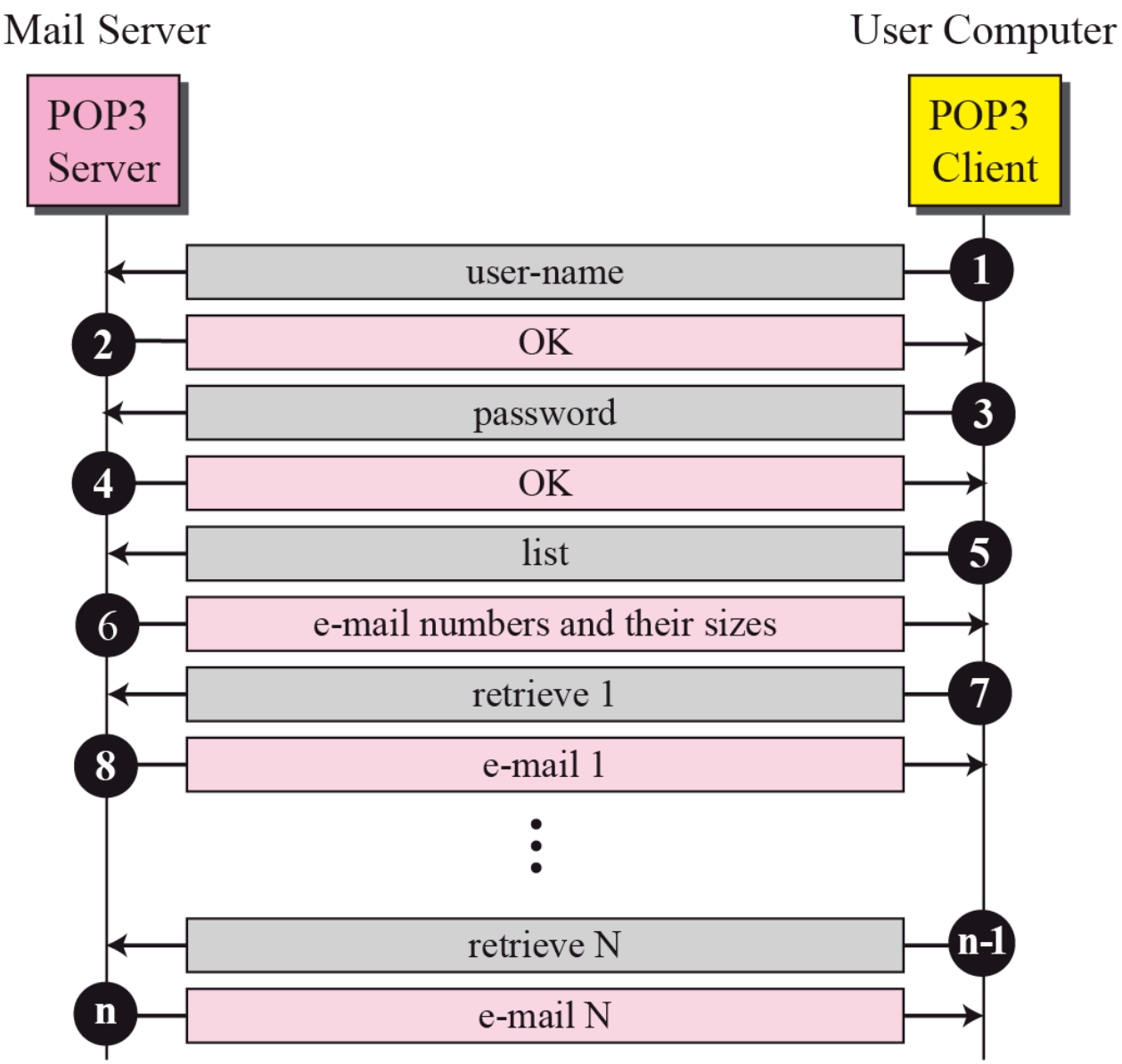
Type	*Subtype*	*Description*
Text	Plain	Unformatted
	HTML	HTML format (see Appendix E)
Multipart	Mixed	Body contains ordered parts of different data types
	Parallel	Same as above, but no order
	Digest	Similar to Mixed, but the default is message/RFC822
	Alternative	Parts are different versions of the 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)

POST OFFICE PROTOCOL VERSION 3 (POP3)

- The **Post Office Protocol** (**POP**) is an application-layer Internet standard **protocol** used by local e-mail clients to retrieve e-mail from a remote server over a TCP/IP connection.
- The **POP (Post Office Protocol 3)** protocol provides a simple, standardized way for users to access mailboxes and download messages to their computers.
- When using the POP protocol all your eMail messages will be downloaded from the mail server to your local computer.
- You can choose to leave copies of your eMails on the server as well.
- The advantage is that once your messages are downloaded you can cut the internet connection and read your eMail at your leisure without incuring further communication costs. On the other hand you might have transferred a lot of message (including spam or viruses) in which you are not at all interested at this point.



- POP supports simple download-and-delete requirements for access to remote mailboxes (termed mail drop in the POP. Although most POP clients have an option to leave mail on server after download, e-mail clients using POP generally connect, retrieve all messages, store them on the user's PC as new messages, delete them from the server, and then disconnect.
- A POP3 server listens on well-known port 110



COMPARISON WITH IMAP:

- POP is a much simpler protocol, making implementation easier.
- POP mail moves the message from the email server onto your local computer, although there is usually an option to leave the messages on the email server as well.
- IMAP defaults to leaving the message on the email server, simply downloading a local copy.
- POP treats the mailbox as one store, and has no concept of folders

- An IMAP client performs complex queries, asking the server for headers, or the bodies of specified messages, or to search for messages meeting certain criteria. Messages in the mail repository can be marked with various status flags (e.g. "deleted" or "answered") and they stay in the repository until explicitly removed by the user—which may not be until a later session..
- The POP protocol requires the currently connected client to be the only client connected to the mailbox. In contrast, the IMAP protocol specifically allows simultaneous access by multiple clients and provides mechanisms for clients to detect changes made to the mailbox by other, concurrently connected, clients.
- When POP retrieves a message, it receives all parts of it, whereas the IMAP4 protocol allows clients to retrieve any of the individual MIME parts separately - for example retrieving the plain text without retrieving attached files.
- IMAP supports flags on the server to keep track of message state: for example, whether or not the message has been read, replied to, or deleted.

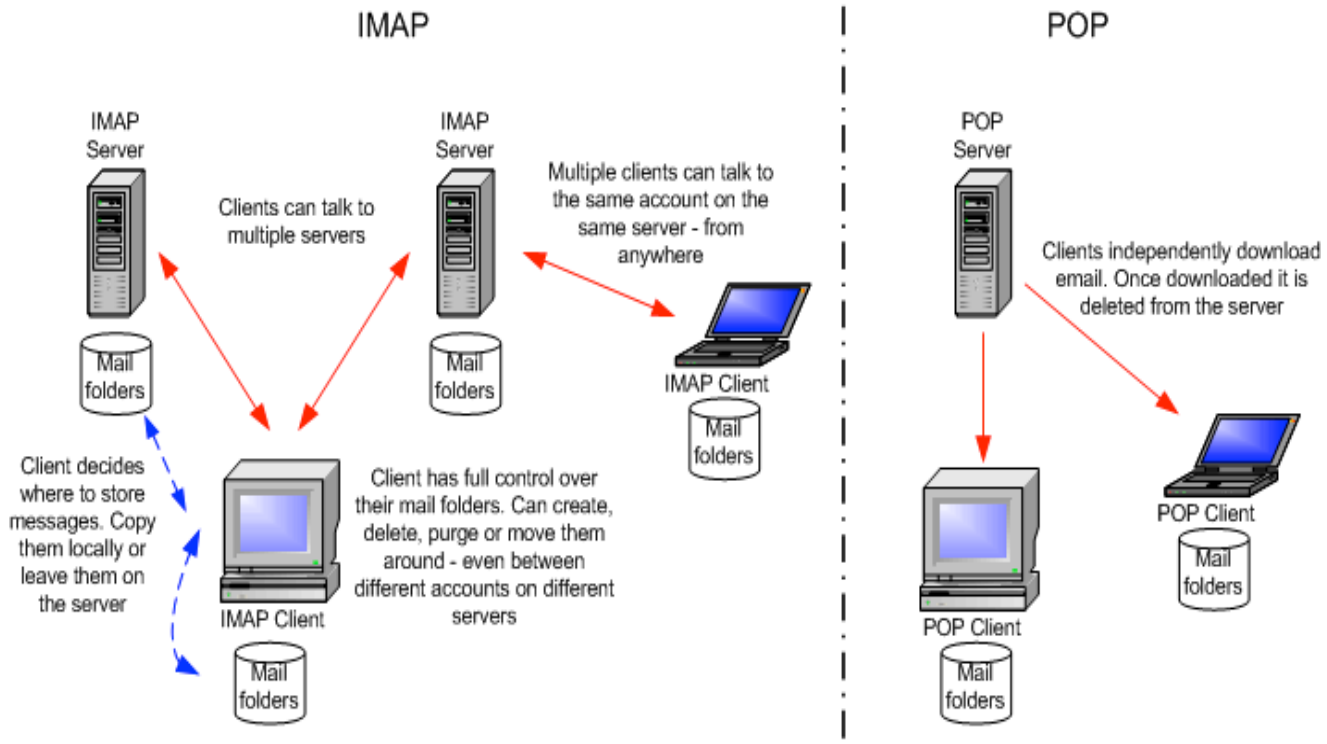
INTERNET MAIL ACCESS PROTOCOL (IMAP)

IMAP (Internet Message Access Protocol) – Is a standard protocol for accessing e-mail from your local server. IMAP is a client/server protocol in which e-mail is received and held for you by your Internet server. As this requires only a small data transfer this works well even over a slow connection such as a modem. Only if you request to read a specific email message will it be downloaded from the server. You can also create and manipulate folders or mailboxes on the server, delete messages etc.

IMAP was designed with the goal of permitting complete management of an email box by multiple email clients; therefore, clients generally leave messages on the server until the user explicitly deletes them. An IMAP server typically listens on port number 143. IMAP over SSL (**IMAPS**) is assigned the port number 993.

IMAP offers you:

- Compatibility with Internet messaging standards, such as 'MIME'
- Access to your messages from multiple computers
- Access to messages without having to download from the server or transfer messages from one computer to another
- Support for "online", "offline", and "disconnected" access modes
- Support for concurrent access to shared mailboxes



THE DIFFERENCES BETWEEN POP AND IMAP:

- An IMAP client synchronizes the e-mail on your computer with the contents of your account on the e-mail server, while a POP account simply downloads the inbox.
- IMAP copies messages from the server instead of deleting them; when you use IMAP, an e-mail message is only removed from the e-mail server when you choose to delete it. Instead of moving messages from the server to your computer, IMAP synchronizes your computer with the e-mail server.

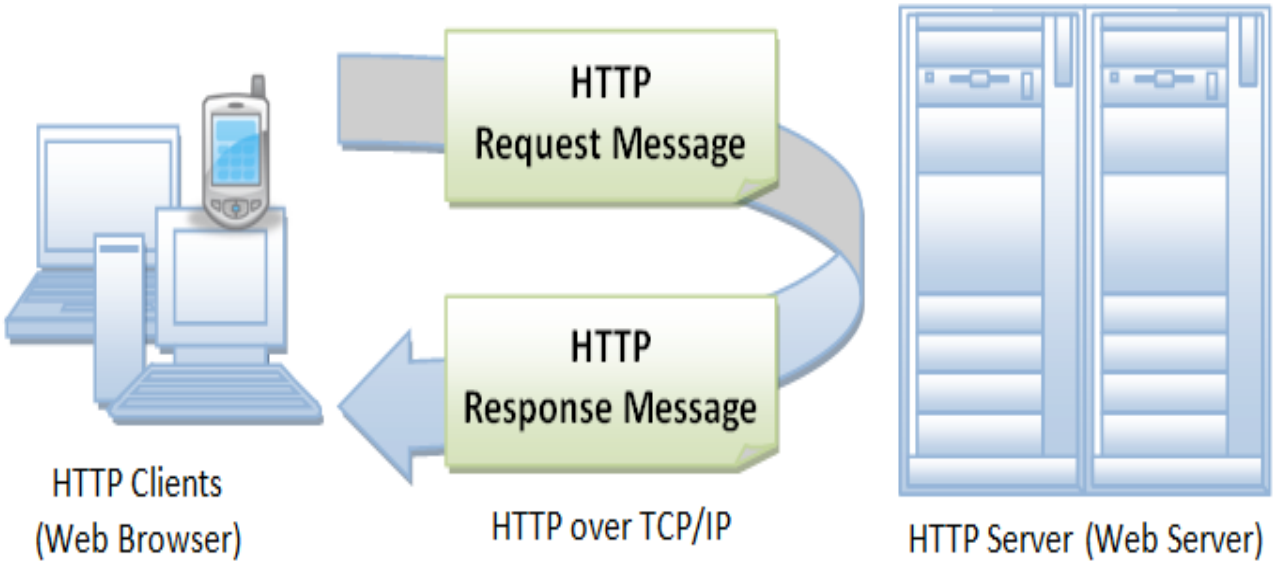
- With a POP account, if you download a message and move it to another folder within your Mail Client, it is still in your webmail, marked as unread.
- If you were to check your e-mail from three different computers via IMAP, all three of those computers, and the e-mail server, would contain all of your e-mail.
- If you chose to delete a message from computer A, it would also be removed from computer B, computer C, and the e-mail server.

Sr.No.	POP3	IMAP4
1	User cannot organize their mail on server.	User Can organize their mail on server.
2	There is only one mailbox (INBOX) exists on the server.	Multiple mailboxes can be created on the desktop PC as well as on the server.
3	A user cannot partially download e-mail.	A user can partially download e-mail.
	All Email will be downloaded into desktop PC if you want to check new mail	A whole message will be downloaded only when it is opened for display from its content

HTTP (HYPERTEXT TRANSFER PROTOCOL)

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.

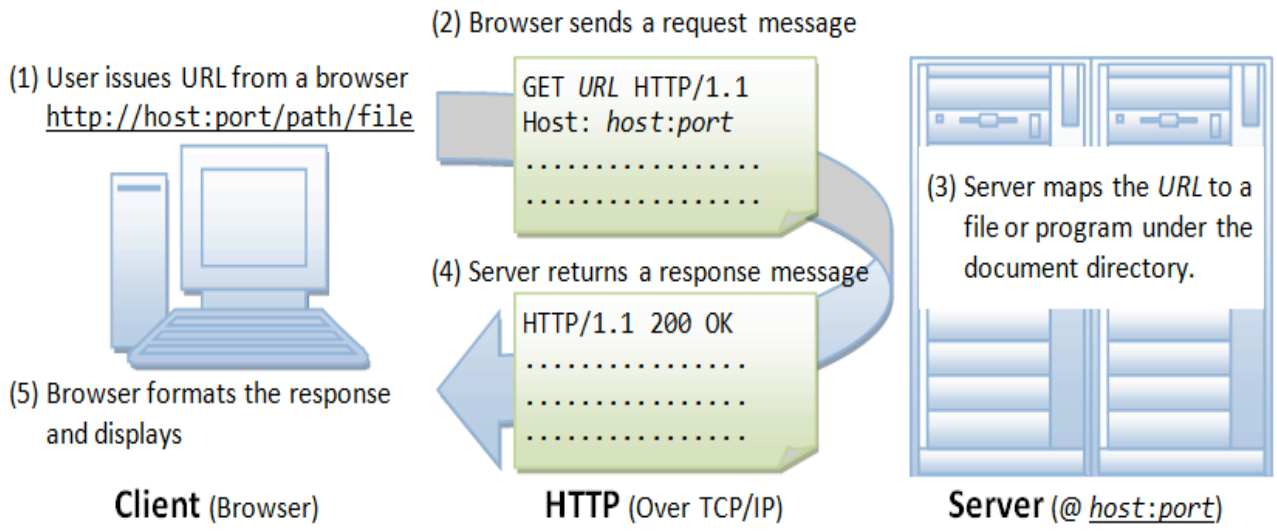
- HTTP (Hypertext Transfer Protocol) is perhaps the most popular application protocol used in the Internet (or The WEB).
- HTTP is an asymmetric request-response client-server protocol as illustrated. An HTTP client sends a request message to an HTTP server. The server, in turn, returns a response message.
- In other words, HTTP is a pull protocol, the client pulls information from the server (instead of server pushes information down to the client).



- HTTP is a stateless protocol. In other words, the current request does not know what has been done in the previous requests.
- HTTP permits negotiating of data type and representation, so as to allow systems to be built independently of the data being transferred.
- Quoting from the RFC2616: "The Hypertext Transfer Protocol (HTTP) is an application-level protocol for distributed, collaborative, hypermedia information systems. It is a generic, stateless, protocol which can be used for many tasks beyond its use for hypertext, such as name servers and distributed object management systems, through extension of its request methods, error codes and headers."

BROWSER:

- Whenever you issue a URL from your browser to get a web resource using HTTP, e.g. http://www.test101.com/index.html, the browser turns the URL into a *request message* and sends it to the HTTP server.
- The HTTP server interprets the request message, and returns you an appropriate response message, which is either the resource you requested or an error message.
- This process is illustrated below:



Uniform Resource Locator (URL)

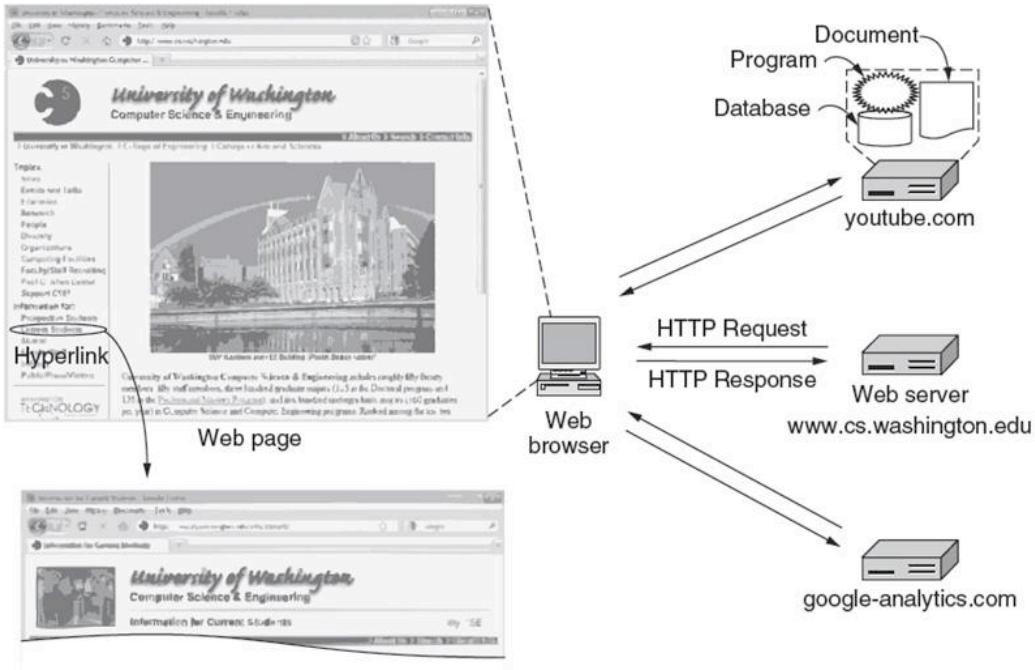
A URL (Uniform Resource Locator) is used to uniquely identify a resource over the web. URL has the following syntax:

protocol://*hostname*:*port*/*path-and-file-name*

There are 4 parts in a URL:

1. ***Protocol*:** The application-level protocol used by the client and server, e.g., HTTP, FTP, and telnet.
2. ***Hostname*:** The DNS domain name (e.g., www.test101.com) or IP address (e.g., 192.128.1.2) of the server.
3. ***Port*:** The TCP port number that the server is listening for incoming requests from the clients.
4. ***Path-and-file-name*:** The name and location of the requested resource, under the server document base directory.

Architectural Overview (1)



Architecture of the Web.

HTTP Protocol:

- As mentioned, whenever you enter a URL in the address box of the browser, the browser translates the URL into a request message according to the specified protocol; and sends the request message to the server.

For example, the browser translated the URL http://www.test101.com/doc/index.html into the following request message:

GET /docs/index.html HTTP/1.1
Host: **www.test101.com**
Accept: image/gif, image/jpeg, */*
Accept-Language: en-us
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)
(blank line)

When this request message reaches the server, the server can take either one of these actions:

1. The server interprets the request received, maps the request into a *file* under the server's document directory, and returns the file requested to the client.
2. The server interprets the request received, maps the request into a *program* kept in the server, executes the program, and returns the output of the program to the client.
3. The request cannot be satisfied, the server returns an error message.

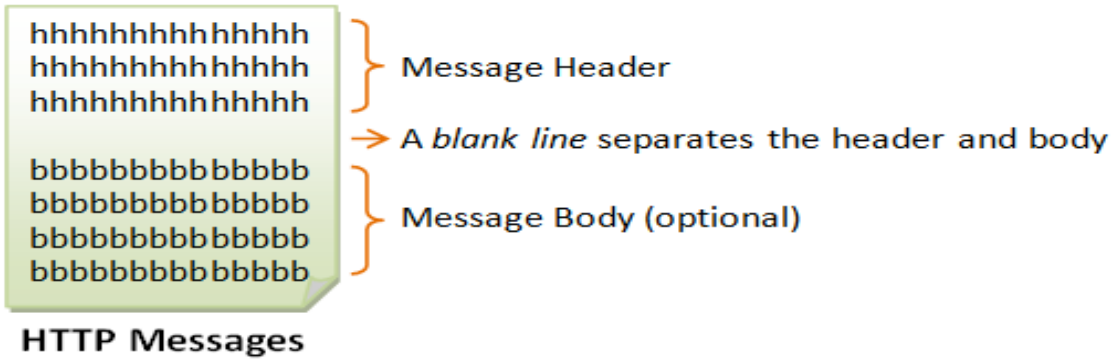
An example of the HTTP response message is as shown:

HTTP/1.1 200 OK
Date: Sun, 18 Oct 2009 08:56:53 GMT
Server: Apache/2.2.14 (Win32)
Last-Modified: Sat, 20 Nov 2004 07:16:26 GMT
ETag: "10000000565a5-2c-3e94b66c2e680"
Accept-Ranges: bytes
Content-Length: 44
Connection: close
Content-Type: text/html
X-Pad: avoid browser bug
<html><body><h1>It works!</h1></body></html>

- The browser receives the response message, interprets the message and displays the contents of the message on the browser's window according to the media type of the response (as in the Content-Type response header). Common media type include "text/plain", "text/html", "image/gif", "image/jpeg", "audio/mpeg", "video/mpeg", "application/msword", and "application/pdf".
- In its idling state, an HTTP server does nothing but listening to the IP address(es) and port(s) specified in the configuration for incoming request.
- When a request arrives, the server analyzes the message header, applies rules specified in the configuration, and takes the appropriate action. The webmaster's main control over the action of web server is via the configuration, which will be dealt with in greater details in the later sections.

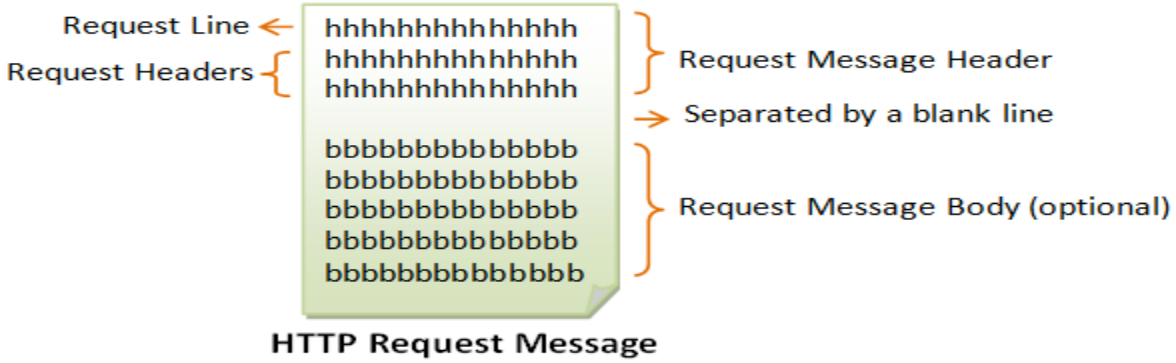
HTTP REQUEST AND RESPONSE MESSAGES:

- HTTP client and server communicate by sending text messages. The client sends a *request message* to the server. The server, in turn, returns a *response message*.
- An HTTP message consists of a *message header* and an optional *message body*, separated by a *blank line*, as illustrated below:



HTTP Request Message

The format of an HTTP request message is as follow:



Request Line

The first line of the header is called the *request line*, followed by optional *request headers*.

The request line has the following syntax:

request-method-name request-URI HTTP-version

- ***request-method-name*:** HTTP protocol defines a set of request methods, e.g., GET, POST, HEAD, and OPTIONS. The client can use one of these methods to send a request to the server.

- ***request-URI*:** specifies the resource requested.
- ***HTTP-version*:** Two versions are currently in use: HTTP/1.0 and HTTP/1.1.

Examples of request line are:

```
GET /test.html HTTP/1.1

HEAD /query.html HTTP/1.0

POST /index.html HTTP/1.1
```

Request Headers

The request headers are in the form of name:value pairs. Multiple values, separated by commas, can be specified.

request-header-name: *request-header-value1*, *request-header-value2*, ...

HTTP REQUEST METHODS:

HTTP protocol defines a set of request methods. A client can use one of these request methods to send a request message to an HTTP server. The methods are:

- **GET:** A client can use the GET request to get a web resource from the server.
- **HEAD:** A client can use the HEAD request to get the header that a GET request would have obtained. Since the header contains the last-modified date of the data, this can be used to check against the local cache copy.
- **POST:** Used to post data up to the web server.
- **PUT:** Ask the server to store the data.
- **DELETE:** Ask the server to delete the data.
- **TRACE:** Ask the server to return a diagnostic trace of the actions it takes.
- **OPTIONS:** Ask the server to return the list of request methods it supports.
- **CONNECT:** Used to tell a proxy to make a connection to another host and simply reply the content, without attempting to parse or cache it. This is often used to make SSL connection through the proxy.
- Other extension methods.

WEB SERVICES

- Web services are web application components.
- Web services can be published, found, and used on the Web.

Web Services take Web-applications to the Next Level

- By using Web services, your application can publish its function or message to the rest of the world.
- Web services use XML to code and to decode data, and SOAP to transport it (using open protocols).
- With Web services, your accounting department's Win 2k server's billing system can connect with your IT supplier's UNIX server.

Web Services have Two Types of Uses

Reusable application-components:

- There are things applications needs very often. So why make these over and over again?
- Web services can offer application-components like: currency conversion, weather reports, or even language translation as services.

Connect existing software:

- Web services can help to solve the interoperability problem by giving different applications a way to link their data.
- With Web services you can exchange data between different applications and different platforms

WSDL

- WSDL stands for Web Services Description Language

- WSDL is an XML-based language for describing Web services.

SOAP

- SOAP stands for Simple Object Access Protocol
- SOAP is an XML based protocol for accessing Web Services.
- SOAP is based on XML

UDDI

- UDDI stands for Universal Description, Discovery and Integration
- UDDI is a directory service where companies can search for Web services.
- UDDI is described in WSDL

RDF

- RDF stands for Resource Description Framework
- RDF is a framework for describing resources on the web
- RDF is written in XML

The WSDL Document Structure

A WSDL document describes a web service using these major elements:

Element	Description
<types>	A container for data type definitions used by the web service
<message>	A typed definition of the data being communicated
<portType>	A set of operations supported by one or more endpoints
<binding>	A protocol and data format specification for a particular port type

The main structure of a WSDL document looks like this:

<definitions>

<types>
  data type definitions.....
</types>

<message>
  definition of the data being communicated....
</message>

<portType>
  set of operations.....
</portType>

<binding>
  protocol and data format specification....
</binding>

</definitions>

A WSDL document can also contain other elements, like extension elements, and a service element that makes it possible to group together the definitions of several web services in one single WSDL document.

WSDL Ports

The **<portType>** element is the most important WSDL element.

It describes a web service, the operations that can be performed, and the messages that are involved.

The <portType> element can be compared to a function library (or a module, or a class) in a traditional programming language.

WSDL Messages

The **<message>** element defines the data elements of an operation.

Each message can consist of one or more parts. The parts can be compared to the parameters of a function call in a traditional programming language.

WSDL Types

The **<types>** element defines the data types that are used by the web service.

For maximum platform neutrality, WSDL uses XML Schema syntax to define data types.

WSDL Bindings

The **<binding>** element defines the data format and protocol for each port type.

WSDL Example

This is a simplified fraction of a WSDL document:

```
<message name="getTermRequest">
  <part name="term" type="xs:string"/>
</message>

<message name="getTermResponse">
  <part name="value" type="xs:string"/>
</message>

<portType name="glossaryTerms">
  <operation name="getTerm">
    <input message="getTermRequest"/>
```

```
    <output message="getTermResponse"/>
  </operation>
</portType>
```

In this example the **<portType>** element defines "glossaryTerms" as the name of a **port**, and "getTerm" as the name of an **operation**.

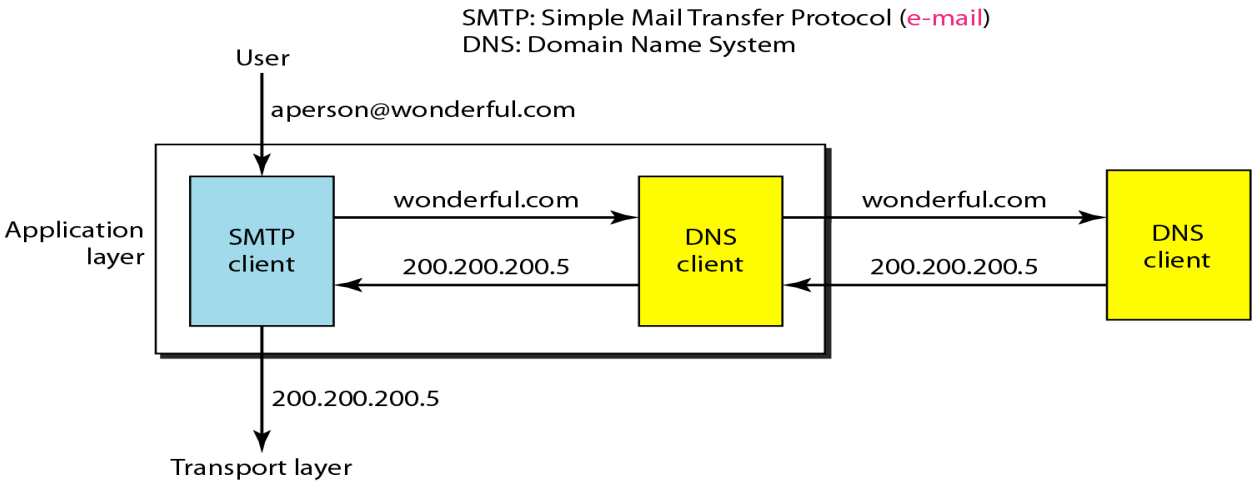
The "getTerm" operation has an **input message** called "getTermRequest" and an **output message** called "getTermResponse".

The **<message>** elements define the **parts** of each message and the associated data types.

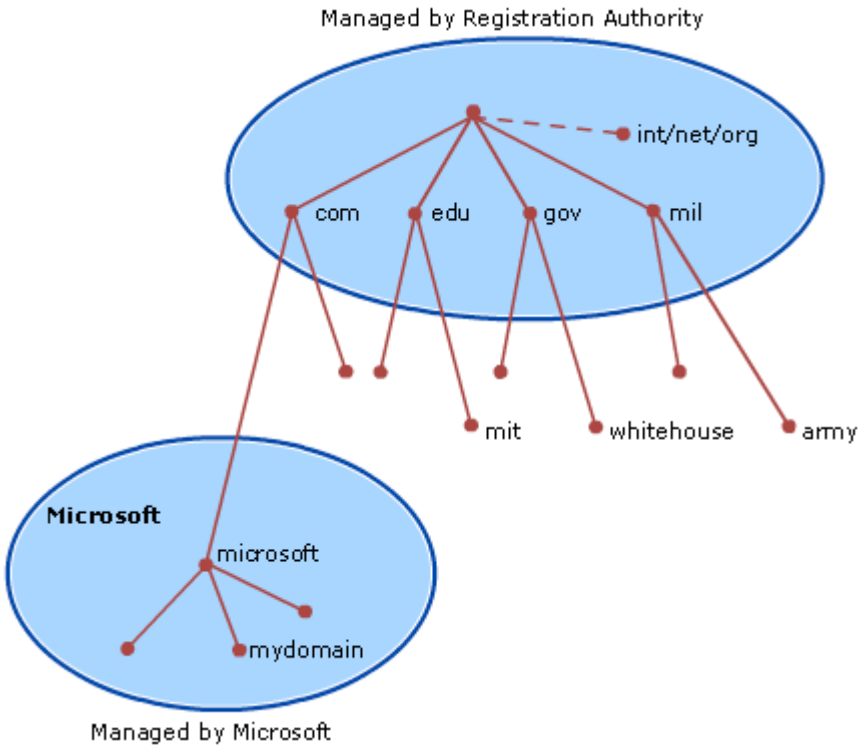
Compared to traditional programming, glossaryTerms is a function library, "getTerm" is a function with "getTermRequest" as the input parameter, and getTermResponse as the return parameter.

DOMAIN NAME SYSTEM

The Domain Name System (**DNS**) is a hierarchical distributed naming system for computers, services, or any resource connected to the Internet or a private network. It associates various information with domain names assigned to each of the participating entities.



DNS Domain Name Hierarchy



Types of DNS Domain Names

Name Type	Description
Root domain	This is the top of the tree, representing an unnamed level; it is sometimes shown as two empty quotation marks (""), indicating a null value. When used in a DNS domain name, it is stated by a trailing period (.) to designate that the name is located at the root or highest level of the domain hierarchy. In this instance, the DNS domain name is considered to be complete and points to an exact location in the tree of names. Names stated this way are called fully qualified domain names (FQDNs).
Top level domain	A name used to indicate a country/region or the type of organization using a name.
Second level domain	Variable-length names registered to an individual or organization for use on the Internet. These names are always based upon an appropriate top-level domain, depending on the type of organization or geographic location where a name is used.
Sub domain	Additional names that an organization can create that are derived from the registered second-level domain name. These include names added to grow the DNS tree of names in an organization and divide it into departments or geographic locations.
Host or resource name	Names that represent a leaf in the DNS tree of names and identify a specific resource. Typically, the leftmost label of a DNS domain name identifies a specific computer on the network. For example, if a name at this level is used in a host (A) RR, it is used to look up the IP address of computer based on its host name.

Some DNS Top-level Domain Names (TLDs)

DNS Domain Name	Type of Organization
com	Commercial organizations
edu	Educational institutions
org	Non-profit organizations
net	Networks (the backbone of the Internet)
gov	Non-military government organizations
mil	Military government organizations
arpa	Reverse DNS
"xx"	Two-letter country code (i.e. us, au, ca, fr

DNS COMPONENTS:

The DNS consists of three components:

- The first is a **"Name Space"** that establishes the syntactical rules for creating and structuring legal DNS names.
- The second is a **"Globally Distributed Database"** implemented on a network of "Name Servers".
- The third is **"Resolver"** software, which understands how to formulate a DNS query and is built into practically every Internet-capable application.

(A) Name Space:

- The DNS “Name Space” is the familiar inverted tree hierarchy with a null node named "" at the top.
- The child nodes of the root node are the Top Level Domains (TLDs)-.com, .net, .org, .gov, .mil-and the country code TLDs, including .jp, .uk, .us, .ca, and so forth. Node names, known as labels, can be as many as 63 characters long, with upper- and lower-case alphabetical letters, numerals, and the hyphen symbol constituting the complete list of legal characters.
- Labels cannot begin with a hyphen. Upper- and lower-case letters are treated equivalently. A label can appear in multiple places within the name space, but no two nodes with the same label can have the same parent node: A node name must be unique among its siblings.

(B) Name Servers:

- The second key component of the DNS is a globally connected network of “name servers”.
- Each zone has a primary or master name server, which is the authoritative source for the zone's resource records.
- The primary name server is the only server that can be updated by means of local administrative activity.
- Secondary or slave name servers hold replicated copies of the primary server's data in order to provide redundancy and reduce the primary server's workload.

Furthermore, name servers generally cache data they have looked up, which can greatly speed up subsequent queries for the same data. Name servers also have a built-in agent mechanism that knows where to ask for data it lacks.

If a name server can't find a domain within its zone, it sends the query a step closer to the root, which will resend it yet a step closer if it can't find the domain itself. The process repeats

until it reaches a TLD, which ensures that the entire depth of the name space will be queried if necessary.

The combination of all the DNS name servers and the architecture of the system creates a remarkable database. There are more than 32 million domain names in the popular TLDs for which the whois utility works. Nominum, whose chief scientist, Paul Mockapetris, invented DNS, claims that there are more than 100 million domain names stored and that the system can easily handle 24,000 queries per second. The database is distributed-no single computer contains all the data. Nevertheless, data is maintained locally even though it's distributed globally, and any device connected to the IP network can perform lookups. The update serial number mechanism in each zone ensures a form of loose coherency on the network-if a record is out of date, the querier knows to check a more authoritative name server.

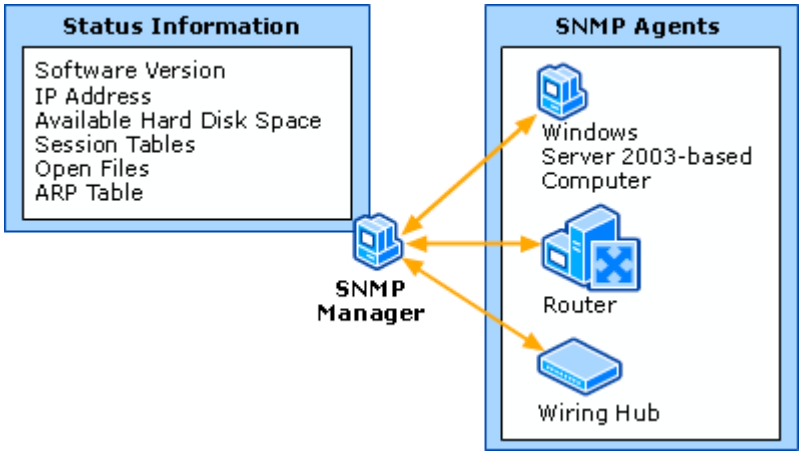
(C) Resolver:

- The third component of the DNS is the “resolver”. The resolver is a piece of software that's implemented in the IP stack of every destination point, or "host" in IETF-speak.
- When a host is configured, manually or through DHCP, it's assigned at least one default name server along with its IP address and subnet mask. This name server is the first place that the host looks in order to resolve a domain name into an IP address.
- If the domain name is in the local zone, the default name server can handle the request. Otherwise, the default name server queries one of the root servers. The root server responds with a list of name servers that contain data for the TLD of the query.
- This response is known as a referral. The name server now queries the TLD name server and receives a list of name servers for the second-level domain name.
- The process repeats until the local name server receives the address for the domain name. The local server then caches the record and returns the address or other DNS data to the original querier.

SIMPLE NETWORK MANAGEMENT PROTOCOL (SNMP)

Simple Network Management Protocol (**SNMP**) is a popular protocol for network management. It is used for collecting information from, and configuring, network devices, such as servers, printers, hubs, switches, and routers on an Internet Protocol (IP) network.

- **Simple Network Management Protocol** (**SNMP**) is an "Internet-standard protocol for managing devices on IP networks".
- Devices that typically support SNMP include routers, switches, servers, workstations, printers, modem racks and more.[1]SNMP is widely used in network management systems to monitor network-attached devices for conditions that warrant administrative attention.



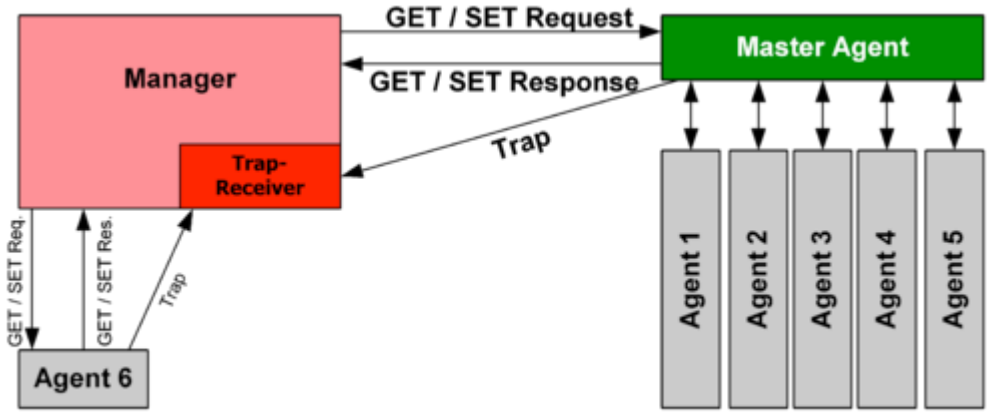
An SNMP-managed network consists of three key components:

- Managed device
- Agent — software which runs on managed devices
- Network management station (NMS) — software which runs on the manager

MANAGED DEVICE:

- A ***managed device*** is network node that implements an SNMP interface that allows unidirectional (read-only) or bidirectional (read and write) access to node-specific information. Managed devices exchange node-specific information with the NMSs.
- Sometimes called network elements, the managed devices can be any type of device, including, but not limited to, routers, access servers, switches, cable modems, bridges, hubs, IP telephones, IP video cameras, computer hosts, and printers.
- An ***agent*** is a network-management software module that resides on a managed device. An agent has local knowledge of management information and translates that information to or from an SNMP-specific form.
- A ***network management station*** **(NMS)** executes applications that monitor and control managed devices. NMSs provide the bulk of the processing and memory resources required for network management. One or more NMSs may exist on any managed network.

Principle of SNMP Communication:



NMP Agents:

SNMP agents do the bulk of the work. They are responsible for gathering information about the local system and storing them in a format that can be queried.updating a database called the "management information base", or **MIB**.

The MIB is a hierarchical, pre-defined structure that stores information that can be queried or set. This is available to well-formed SNMP requests originating from a host that has authenticated with the correct credentials (an SNMP manager).

The agent computer configures which managers should have access to its information. It can also act as an intermediary to report information on devices it can connect to that are not configured for SNMP traffic. This provides a lot of flexibility in getting your components online and SNMP accessible.

SNMP agents respond to most of the commands defined by the protocol. These include GetRequest,GetNextRequest, GetBulkRequest, SetRequest and InformRequest. In addition, an agent is designed to send Trap messages.

Management information base (MIB)

- The most difficult part of the SNMP system to understand is probably the **MIB**, or management information base.

- The MIB is a database that follows a standard that the manager and agents adhere to. It is a hierarchical structure that, in many areas, is globally standardized, but also flexible enough to allow vendor-specific additions.

- The MIB structure is best understood as a top-down hierarchical tree. Each branch that forks off is labeled with both an identifying number (starting with 1) and an identifying string that are unique for that level of the hierarchy. You can use the strings and numbers interchangeably.

To refer to a specific node of the tree, you must trace the path from the unnamed root of the tree to the node in question. The lineage of its parent IDs (numbers or strings) are strung together, starting with the most general, to form an address. Each junction in the hierarchy is represented by a dot in this notation, so that the address ends up being a series of ID strings or numbers separated by dots. This entire address is known as an object identifier, or **OID**.

Hardware vendors that embed SNMP agents in their devices sometimes implement custom branches with their own fields and data points. However, there are standard MIB branches that are well defined and can be used by any device.

The standard branches we will be discussing will all be under the same parent branch structure. This branch defines information that adheres to the MIB-2 specification, which is a revised standard for compliant devices.