CN: WEEK 3

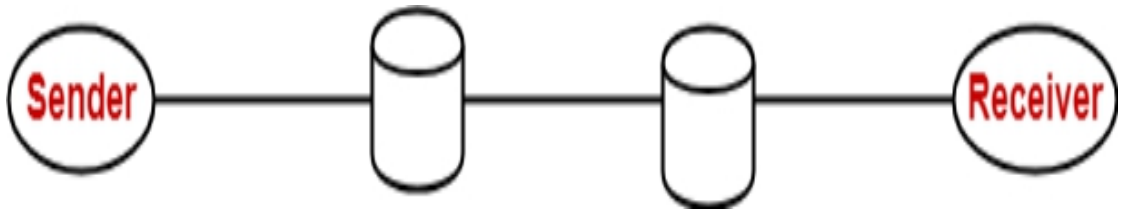
Transmission delay dominates Propagation delay in Packet switching

- While calculating the total time, we often ignore the propagation delay.
- The reason is in packet switching, transmission delay dominates over propagation delay.
- This is because each packet is transmitted over the link at each hop.

Packet switching numerical example II

In a packet switching network, packets are routed from source to destination along a single path having two intermediate nodes. If the message size is 24 bytes and each packet contains a header of 3 bytes, then the optimum packet size is-

4 bytes
6 bytes
9 bytes



(Let bandwidth of the network = X Bps and 1 / X = a)

Option-A: Packet Size = 4 Bytes

Here,

- The entire message is divided into packets of size 4 bytes.
- These packets are then sent one after the other.

Data size = Packet size – Header size
= 4 bytes – 3 bytes = 1 byte

Thus, 1 byte of data can be sent in each packet.

Number of packets required
= Total data to be sent / Data contained in one packet
= 24 bytes / 1 byte = 24 packets

Transmission delay
= Packet size / Bandwidth
= 4 bytes / X Bps = 4a sec

Time taken by the first packet to reach from sender to receiver
= 3 x Transmission delay = 3 x 4a sec = 12a sec

Time taken by the remaining packets to reach from sender to receiver
= Number of remaining packets x Transmission delay
= 23 x 4a sec = 92a sec

Total time taken to send the complete message from sender to receiver
= 12a sec + 92a sec = 104a sec

Option-B: Packet Size = 6 Bytes

Here, the entire message is divided into packets of size 6 bytes.
These packets are then sent one after the other.

Data size = Packet size – Header size
= 6 bytes – 3 bytes = 3 bytes

Thus, only 3 bytes of data can be sent in each packet.

Number of packets required
= Total data to be sent / Data contained in one packet
= 24 bytes / 3 bytes = 8 packets

Transmission delay
= Packet size / Bandwidth
= 6 bytes / X Bps = 6a sec

Time taken by the first packet to reach from sender to receiver
= 3 x Transmission delay = 3 x 6a sec = 18a sec

Time taken by the remaining packets to reach from sender to receiver
= No of remaining packets x Transmission delay
= 7 x 6a sec = 42a sec

Total time taken to send the complete message from sender to receiver = 18a sec + 42a sec = 60a sec

Option-B: Packet Size = 9 Bytes

Here, the entire message is divided into packets of size 9 bytes. These packets are then sent one after the other.

Data size = Packet size – Header size = 9 – 3 = 6 bytes
Thus, only 6 bytes of data can be sent in each packet.

Number of packets required
= Total data to be sent / Data contained in one packet
= 24 bytes / 6 bytes = 4 packets

Transmission delay
= Packet size / Bandwidth = 9 bytes / X Bps = 9a sec

Time taken by the first packet to reach from sender to receiver = 3 x Transmission delay = 3 x 9a sec = 27a sec

Time taken by the remaining packets to reach from sender to receiver = Number of remaining packets x Transmission delay = 3 x 9a sec = 27a sec

Total time taken to send the complete message from sender to receiver = 27a sec + 27a sec = 54a sec

From here,

Total time taken when packet size is 4 bytes = 104a sec
Total time taken when packet size is 6 bytes = 60a sec
Total time taken when packet size is 9 bytes = 54a sec

Result:

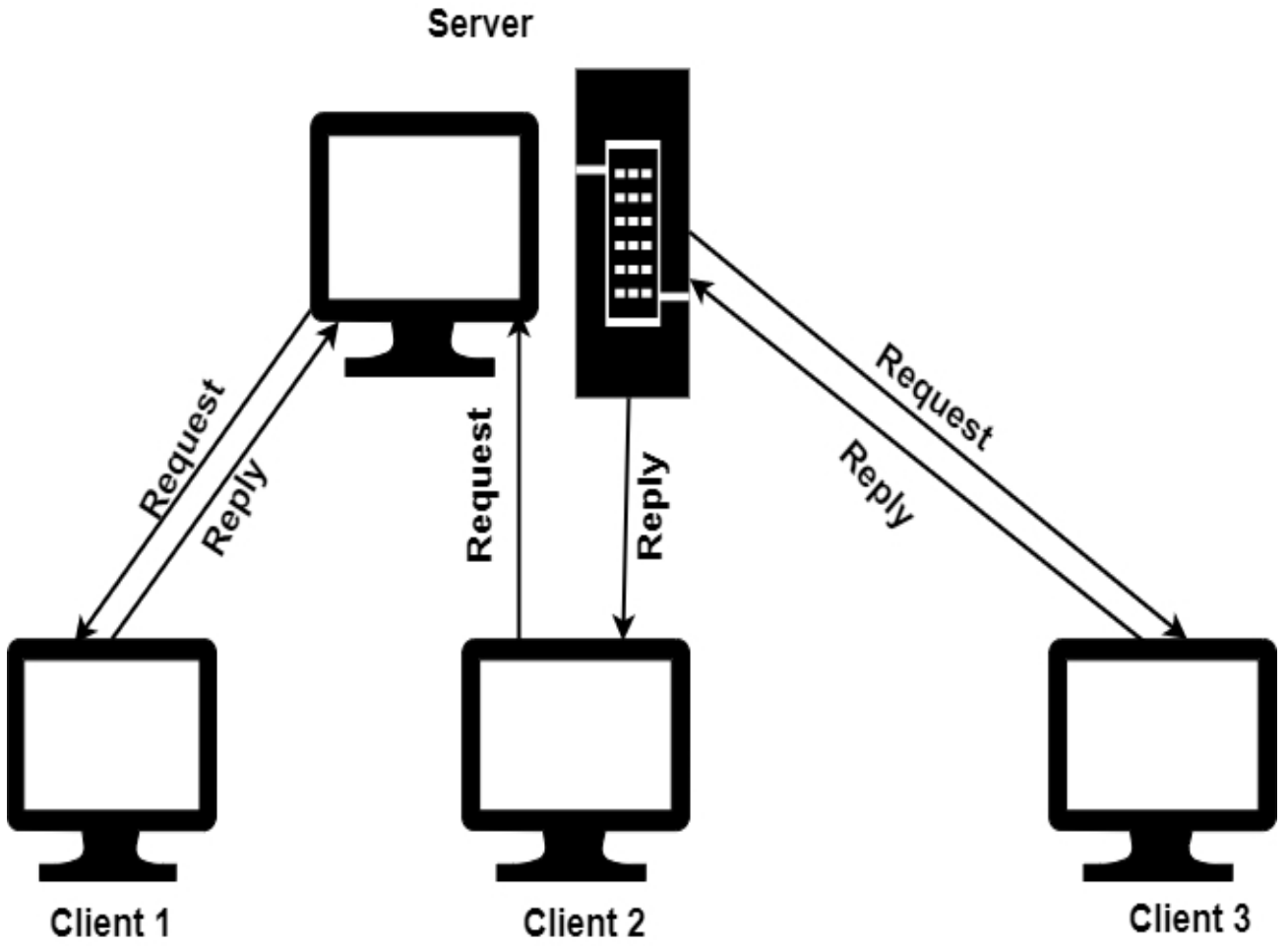
Time taken is minimum when packet size is **9 bytes.**

Client-Server Architecture

The Client-server model is a distributed application structure that partitions task or workload between the providers of a resource or service, called servers, and service requesters called clients.

In the client-server architecture, when the client computer sends a request for data to the server through the internet, the server accepts the requested process and deliver the data packets requested back to the client.

Examples of Client-Server Model are Email, World Wide Web, etc



Client Server Model

Server-side Programming :

It is the program that runs on server dealing with the generation of content of web page.

1) Querying the database
2) Operations over databases
3) Access/Write a file on server.
4) Interact with other servers.
5) Structure web applications.
6) Process user input. (For example if user input is a text in search box, run a search algorithm on data stored on server and send the results.)

The Programming languages for server-side programming are :
1) PHP
2) C++
3) Java and JSP
4) Python
5) Ruby on Rails

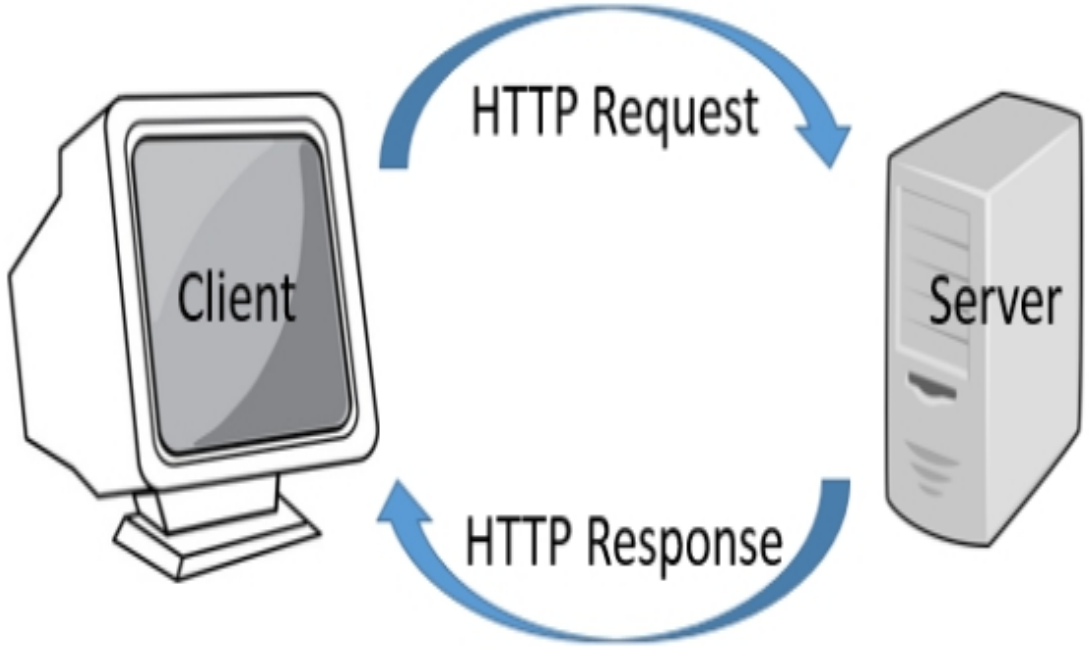
Client-side Programming :

It is the program that runs on the client machine (browser) and deals with the user interface/display and any other processing that can happen on client machine like reading/writing cookies.

1) Interact with temporary storage
2) Make interactive web pages
3) Interact with local storage
4) Sending request for data to server
5) Send request to server
6) work as an interface between server and user

The Programming languages for client-side programming are :
1) Javascript
2) VBScript
3) HTML
4) CSS
5) AJAX

HTTP (Hyper Text Transfer Protocol)



HTTP is application-level protocol for collaborative, distributed, hypermedia information systems.

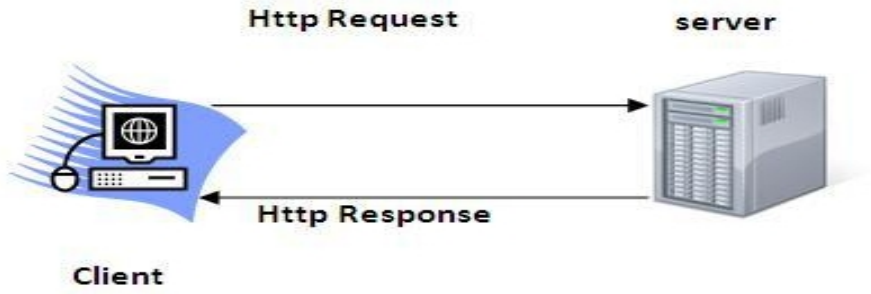
It is the data communication protocol used to establish communication between client and server.

It is TCP/IP based communication protocol, which is used to deliver the data like image files, query results, HTML files etc on the World Wide Web (WWW) with the default port is TCP 80.

It provides the standardized way for computers to communicate with each other.

It is the protocol that allows web servers and browsers to exchange data over the web.

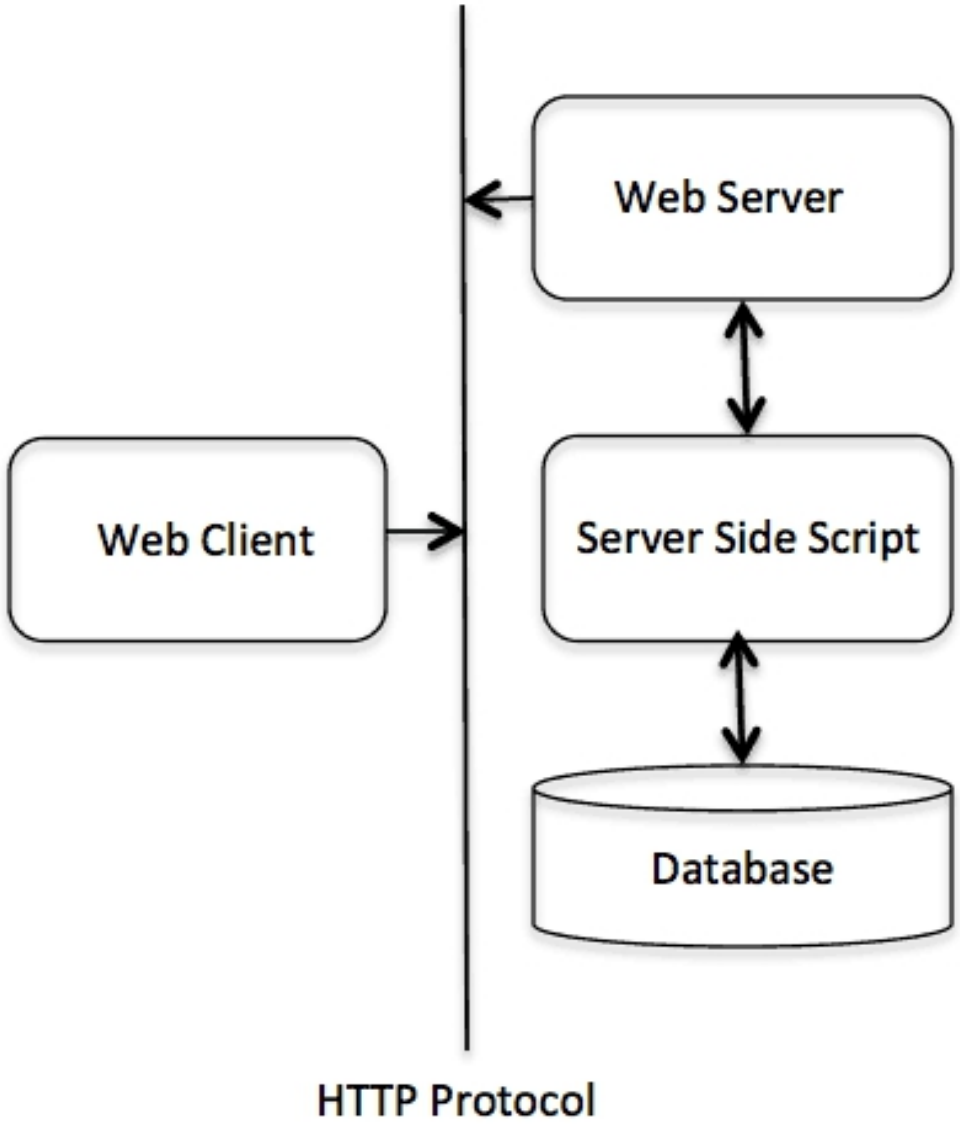
It is a request response protocol.



HTTP is media independent: It specifies that any type of media content can be sent by HTTP as long as both the server and the client can handle the data content.

HTTP is connectionless: It is a connectionless approach in which HTTP client i.e., a browser initiates the HTTP request and after the request is sent the client disconnects from server and waits for the response.

HTTP is stateless: The client and server are aware of each other during a current request only. Afterwards, both of them forget each other. Due to the stateless nature of protocol, neither the client nor the server can retain the information about different request across the web pages.



HTTP is request/response protocol which is based on client/server based architecture. In this protocol, web browser, search engines, etc. behave as HTTP clients and the Web server like Servlet behaves as a server

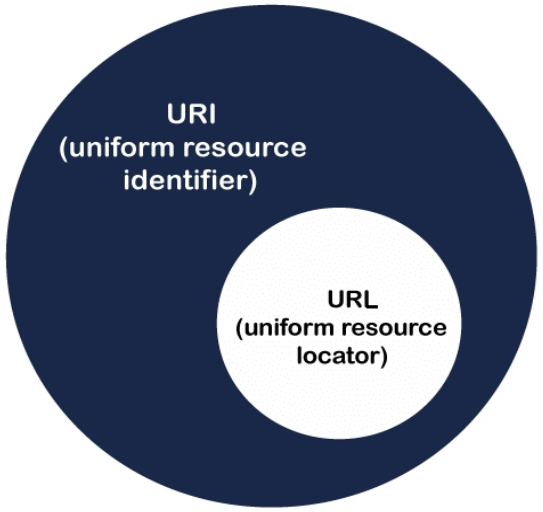
HTTP Requests

The request sent by the computer to a web server, contains all sorts of potentially interesting information; it is known as ***HTTP requests.***

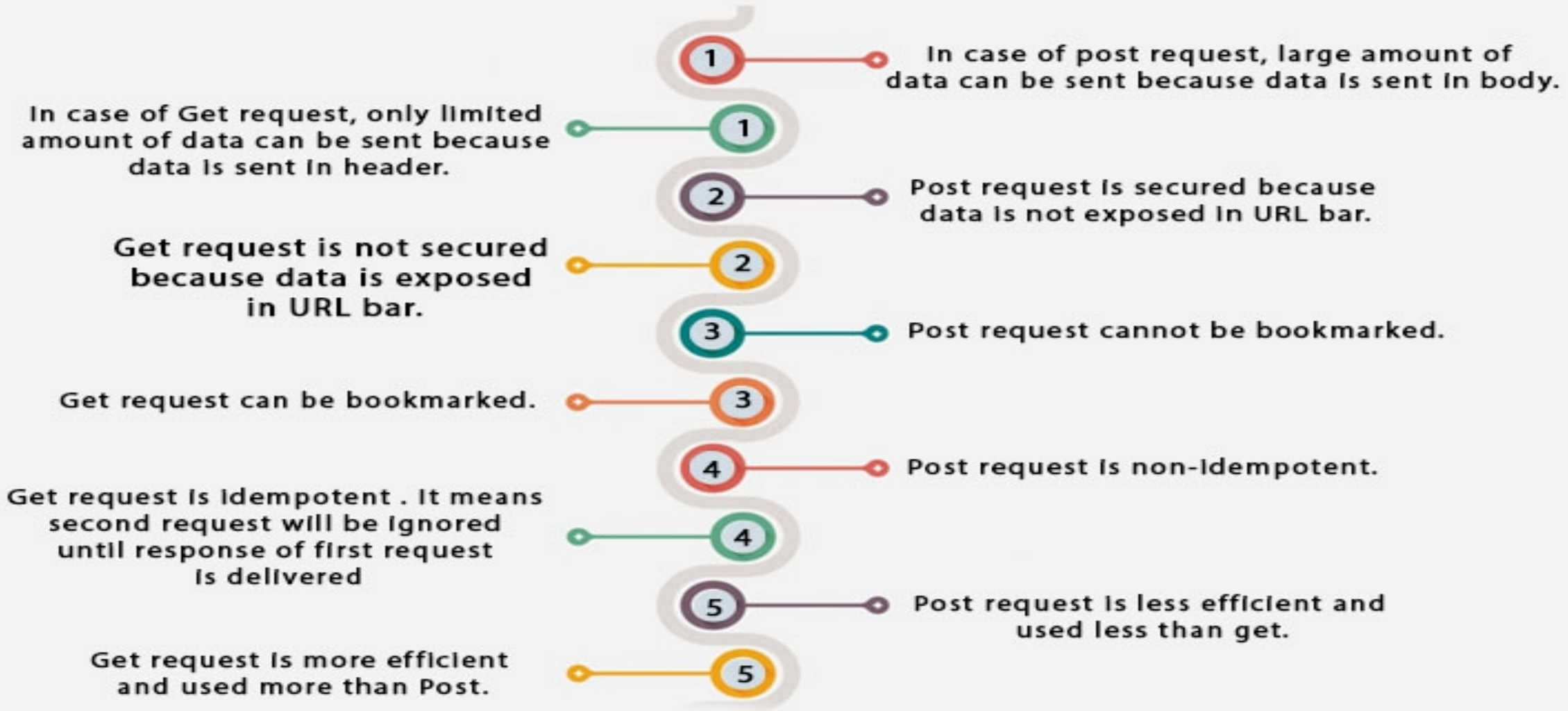
The HTTP request method indicates the method to be performed on the resource identified by the Requested URI (Uniform Resource Identifier).

HTTP Request	Description
GET	Asks to get the resource at the requested URL.
POST	Asks the server to accept the body info attached. It is like GET request with extra info sent with the request.
HEAD	Asks for only the header part of whatever a GET would return. Just like GET but with no body.
TRACE	Asks for the loopback of the request message, for testing or troubleshooting.
PUT	Says to put the enclosed info (the body) at the requested URL.
DELETE	Says to delete the resource at the requested URL.
OPTIONS	Asks for a list of the HTTP methods to which the thing at the request URL can respond

URI	URL
URI is an acronym for Uniform Resource Identifier.	URL is an acronym for Uniform Resource Locator.
URI contains two subsets, URN, which tell the name, and URL, which tells the location.	URL is the subset of URI, which tells the only location of the resource.
All URIs cannot be URLs, as they can tell either name or location.	All URLs are URIs, as every URL can only contain the location.
A URI aims to identify a resource and differentiate it from other resources by using the name of the resource or location of the resource.	A URL aims to find the location or address of a resource on the web.
An example of a URI can be ISBN 0-486-35557-4.	An example of an URL is https://www.javatpoint.com.
It is commonly used in XML and tag library files such as JSTL and XSTL to identify the resources and binaries.	It is mainly used to search the webpages on the internet.
The URI scheme can be protocol, designation, specification, or anything.	The scheme of URL is usually a protocol such as HTTP, HTTPS, FTP, etc.



Get vs. Post


1
In case of post request, large amount of data can be sent because data is sent in body.
In case of Get request, only limited amount of data can be sent because data is sent in header.
1
2
Post request is secured because data is not exposed in URL bar.
Get request is not secured because data is exposed in URL bar.
2
3
Post request cannot be bookmarked.
Get request can be bookmarked.
3
4
Post request is non-idempotent.
Get request is idempotent . It means second request will be ignored until response of first request is delivered
4
5
Post request is less efficient and used less than get.
Get request is more efficient and used more than Post.
5

Imp features of GET requests are:

- *It remains in the browser history*
- *It can be bookmarked*
- *It can be cached*
- *It have length restrictions*
- *It should never be used when dealing with sensitive data*
- *It should only be used for retrieving the data*

Imp features of POST requests are:

- *This requests cannot be bookmarked*
- *This requests have no restrictions on length of data*
- *This requests are never cached*
- *This requests do not retain in the browser history*

HTTP - Messages

HTTP is based on the client-server architecture model and a stateless request/response protocol that operates by exchanging messages across a reliable TCP/IP connection.

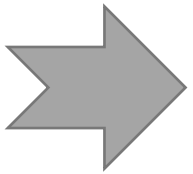
An HTTP "client" is a program that establishes a connection to a server for the purpose of sending one or more HTTP request messages. An HTTP "server" is a program that accepts connections in order to serve HTTP requests by sending HTTP response messages.

HTTP makes use of the Uniform Resource Identifier (URI) to identify a given resource and to establish a connection. Once the connection is established, HTTP messages are passed in a format similar to that used by the Internet mail and the Multipurpose Internet Mail Extensions.

These messages include requests from client to server and responses from server to client which will have the following format:

```
HTTP-message   = <Request> | <Response> ; HTTP/1.1 messages
```

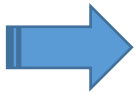
HTTP requests and HTTP responses use a generic message format for transferring the required data. This generic message format consists of the following four items.



- A Start-line
- Zero or more header fields followed by CRLF
- An empty line (i.e., a line with nothing preceding the CRLF) indicating the end of the header fields
- Optionally a message-body

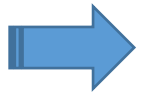
Message Start-Line:

A start-line will have the following generic syntax:



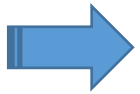
```
start-line = Request-Line | Status-Line
```

A request-line consists of three parts: a method name, requested resource's local path, and the HTTP version being used.



```
GET /path/to/file/index.html HTTP/1.0
```

A Status-line has three parts: the HTTP version, a response status code that gives the result of the request, and the English reason phrase describing the status code.



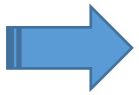
```
HTTP/1.0 200 OK
or
HTTP/1.0 404 Not Found
```

Message Headers

The Message header provides information about the request and response. It also provides information about the object which is sent in the message body. Message Headers are of four types:

- *General Header:* It has general applicability for both request messages and response messages.
- *Request Header:* It has applicability only for the request messages.
- *Response Header:* It has applicability only for the response messages.
- *Entity Header:* It defines meta-information about the entity-body, and about the resource identified by request.

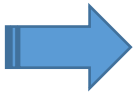
All the above headers follow the same generic format.



message-header = field-name ":" [field-value]

Message Body

The message body of an HTTP message is used to carry the entire body associated with the request and response.



```
message-body = entity-body
             | <entity-body encoded as per Transfer-Encoding>
```

Transfer-Encoding MUST be used to indicate any transfer-codings which is applied by an application to ensure safe and proper transfer of the message. Transfer-Encoding is a property of the message.


MAC Address
Vs
IP Address

Both MAC Address and IP Address are used to uniquely defines a device on the internet. NIC Card's Manufacturer provides the MAC Address, on the other hand Internet Service Provider provides IP Address.

The main difference between MAC and IP address is that, MAC Address is used to ensure the physical address of computer. It uniquely identifies the devices on a network. While IP address are used to uniquely identifies the connection of network with that device that take part in a network.

Why need Mac address?

- It provides a secure way to find senders or receivers in the network.
- MAC address helps you to prevent unwanted network access.
- MAC address is a unique number; hence it can be used to track the device.
- Wi-Fi networks at the airport use the MAC address of a specific device to identify it.

Why need an IP address?

- An IP address is assigned to every device on a network so that the device can be located on that network.
- It helps you to develop a virtual connection between a destination and a source.
- IP address is one type of numerical label assigned to each device connected to a computer network that uses the IP for communication.
- It acts as an identifier for a specific machine on a particular network.
- It helps you to specify the technical format of the addressing and packaging scheme.

MAC Address stands for Media Access Control Address.	IP Address stands for Internet Protocol Address.
MAC Address is a six byte hexadecimal address.	IP Address is either four byte (IPv4) or eight byte (IPv6) address.
A device attached with MAC Address can retrieve by ARP protocol.	A device attached with IP Address can retrieve by RARP protocol.
NIC Card's Manufacturer provides the MAC Address.	Internet Service Provider provides IP Address.
MAC Address is used to ensure the physical address of computer.	IP Address is the logical address of the computer.
MAC Address operates in the data link layer.	IP Address operates in the network layer.
MAC Address helps in simply identifying the device.	IP Address identifies the connection of the device on the network.
MAC Address of computer cannot be changed with time and environment.	IP Address modifies with the time and environment.
MAC Address can't be found easily by third party.	IP Address can be found by third party.


DNS

Domain Name System (DNS)

- DNS is a host name to IP address translation service.
- It is an application layer protocol for message exchange between clients and servers.
- Each device connected to Internet has a unique IP address which other machines use to find the device. DNS servers eliminate the need for humans to memorize IP addresses such as 192.168.1.1

Why DNS ??

Every host is identified by the IP address but remembering numbers is very difficult for the people and also the IP addresses are not static therefore a mapping is required to change the domain name to IP address. So DNS is used to convert the domain name of the websites to their numerical IP address.

There are 4 DNS servers involved in loading a webpage:

DNS recursor - The recursor can be thought of as a librarian who is asked to go find a particular book somewhere in a library. The DNS recursor is a server designed to receive queries from client machines through applications such as web browsers. Typically the recursor is then responsible for making additional requests in order to satisfy the client's DNS query.

Root nameserver - The root server is the first step in translating (resolving) human readable host names into IP addresses. It can be thought of like an index in a library that points to different racks of books - typically it serves as a reference to other more specific locations.

TLD nameserver - The top level domain server (TLD) can be thought of as a specific rack of books in a library. This nameserver is the next step in the search for a specific IP address, and it hosts the last portion of a hostname (In example.com, the TLD server is "com").

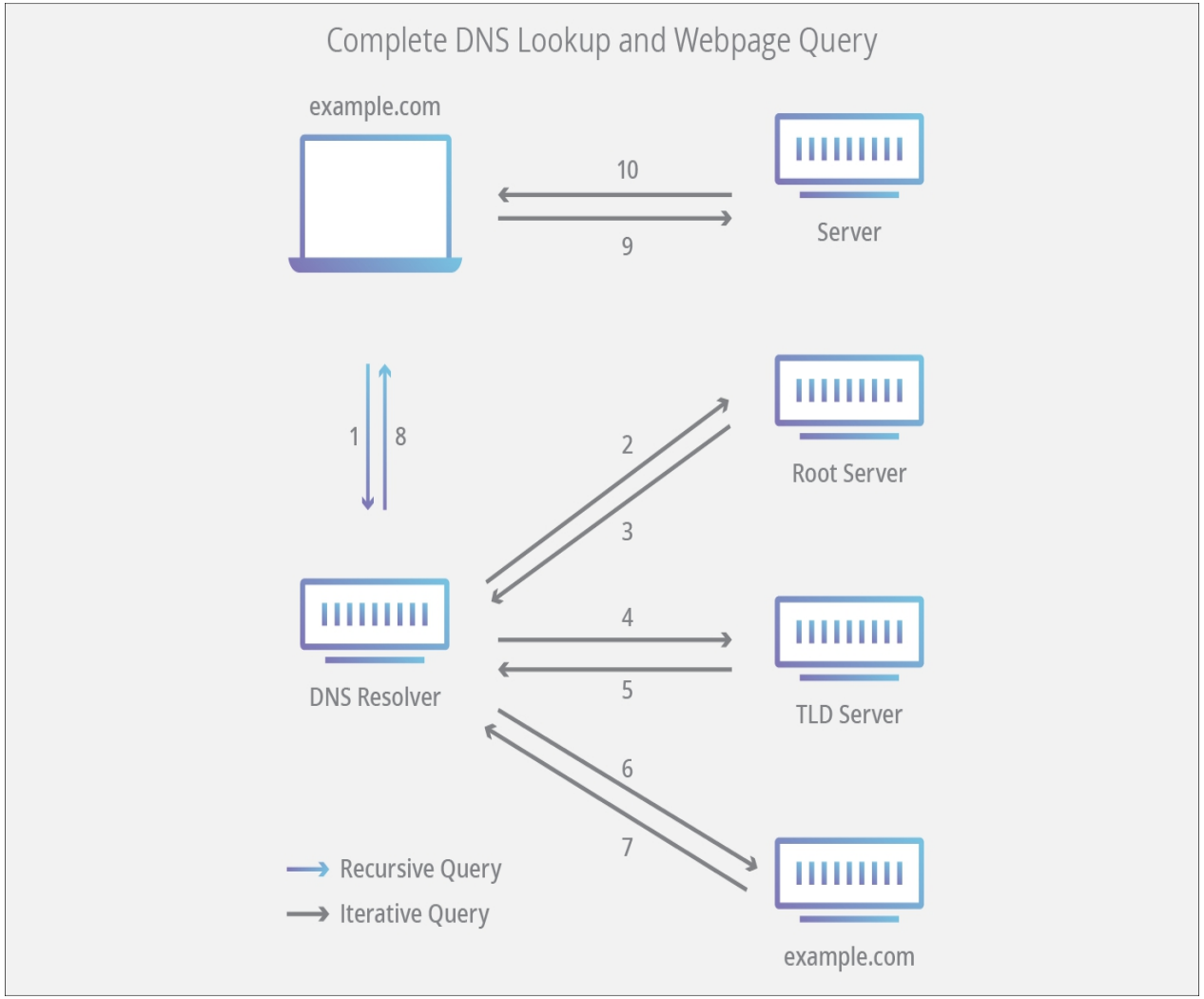
Authoritative nameserver - This final nameserver can be thought of as a dictionary on a rack of books, in which a specific name can be translated into its definition. The authoritative nameserver is the last stop in the nameserver query. If the authoritative name server has access to the requested record, it will return the IP address for the requested hostname back to the DNS Recursor (the librarian) that made the initial request.

8 steps in a DNS

1. A user types ‘example.com’ into a web browser and the query travels into the Internet and is received by a DNS recursive resolver.
2. The resolver then queries a DNS root nameserver (.).
3. The root server then responds to the resolver with the address of a Top Level Domain (TLD) DNS server (such as .com or .net), which stores the information for its domains. When searching for example.com, our request is pointed toward the .com TLD.
4. The resolver then makes a request to the .com TLD.
5. The TLD server then responds with the IP address of the domain’s nameserver, example.com.
6. Lastly, the recursive resolver sends a query to the domain’s nameserver.
7. The IP address for example.com is then returned to the resolver from the nameserver.
8. The DNS resolver then responds to the web browser with the IP address of the domain requested initially.

Once these 8 steps of the DNS lookup have returned the IP address for example.com, the browser is able to make the request for the web page:

- The browser makes a HTTP request to the IP address.
- The server at that IP returns the webpage to be rendered in the browser (step 10).



DNS records

These are instructions that live in authoritative DNS servers and provide information about a domain including what IP address is associated with that domain and how to handle requests for that domain.

- ✓ **A record -** The record that holds the IP address of a domain. Learn more about the A record.
- ✓ **CNAME record -** Forwards one domain or subdomain to another domain, does NOT provide an IP address. Learn more about the CNAME record.
- ✓ **MX record -** Directs mail to an email server. Learn more about the MX record.
- ✓ **TXT record -** Lets an admin store text notes in the record. Learn more about the TXT record.
- ✓ **NS record -** Stores the name server for a DNS entry. Learn more about the NS record.
- ✓ **SOA record -** Stores admin information about a domain. Learn more about the SOA record.
- ✓ **SRV record -** Specifies a port for specific services. Learn more about the SRV record.
- ✓ **PTR record -** Provides a domain name in reverse-lookups. Learn more about the PTR record.

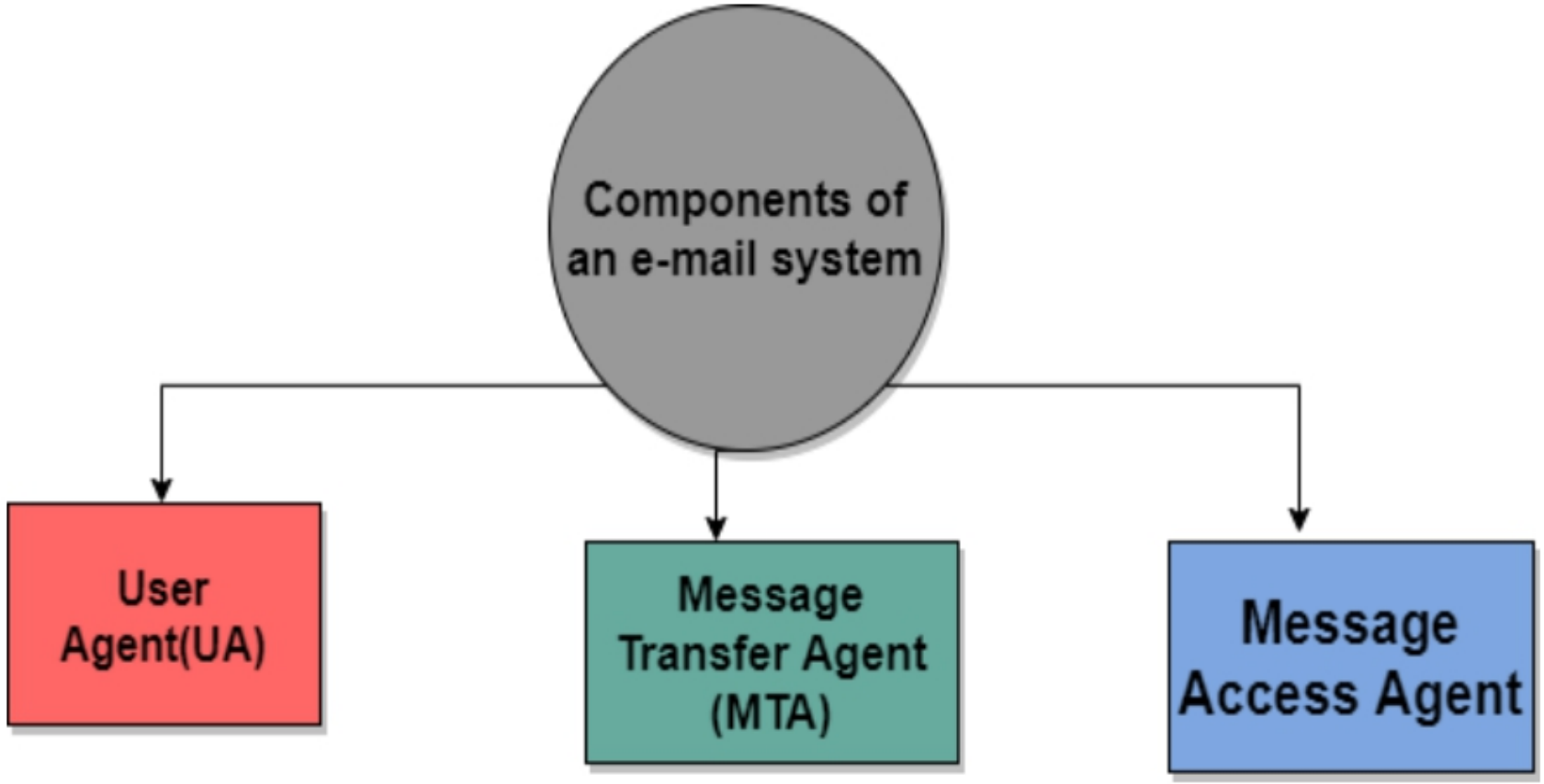

EMAIL

- Electronic mail is often referred to as E-mail and it is a method used for exchanging digital messages.
- Electronic mail is mainly designed for human use.
- It allows a message to includes text, image, audio as well as video.
- This service allows one message to be sent to one or more than one recipient.
- The E-mail systems are mainly based on the store-and-forward model where the E-mail server system accepts, forwards, deliver and store the messages on behalf of users who only need to connect to the infrastructure of the Email.
- The Person who sends the email is referred to as the Sender while the person who receives an email is referred to as the Recipient.

Need of an Email

- We can send the same message to several peoples at the same time.
- It is a very fast and efficient way of transferring information.
- The email system is very fast as compared to the Postal system.
- Information can be easily forwarded to coworkers without retyping it.

Components of E-mail System



User Agent(UA)

It is a program mainly used to send and receive an email.

It is the first component of an Email.

User-agent also handles the mailboxes.

The User-agent mainly provides the services to the user in order to make the sending and receiving process of message easier.

Services provided by the UA:

- *Reading the Message*
- *Replying the Message*
- *Composing the Message*
- *Forwarding the Message.*
- *Handling the Message.*

Message Transfer Agent (MTA)

The actual process of transferring the email is done through the Message Transfer Agent(MTA).

In order to send an Email, a system must have an MTA client.

In order to receive an email, a system must have an MTA server.

The protocol that is mainly used to define the MTA client and MTA server on the internet is called SMTP(Simple Mail Transfer Protocol).

The SMTP mainly defines how the commands and responses must be sent back and forth

Message Access Agent (MAA)

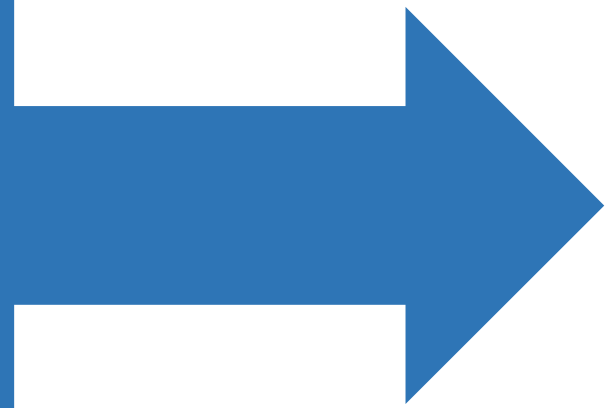
In the first and second stages of email delivery, we make use of SMTP.

SMTP is basically a Push protocol.

The third stage of the email delivery mainly needs the pull protocol, and at this stage, the message access agent is used.

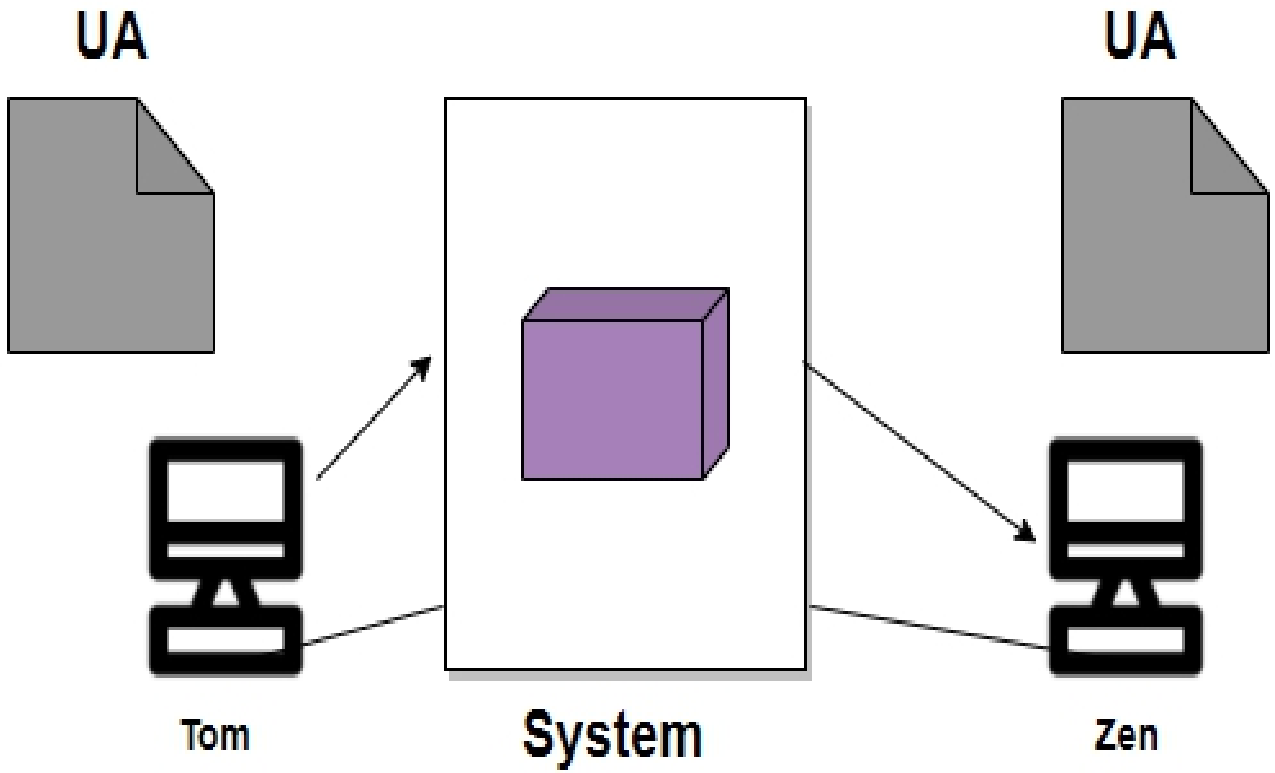
The two protocols used to access messages are POP and IMAP4.

Now its time to take a look at the architecture of e-mail with the help of four scenarios:



First Scenario

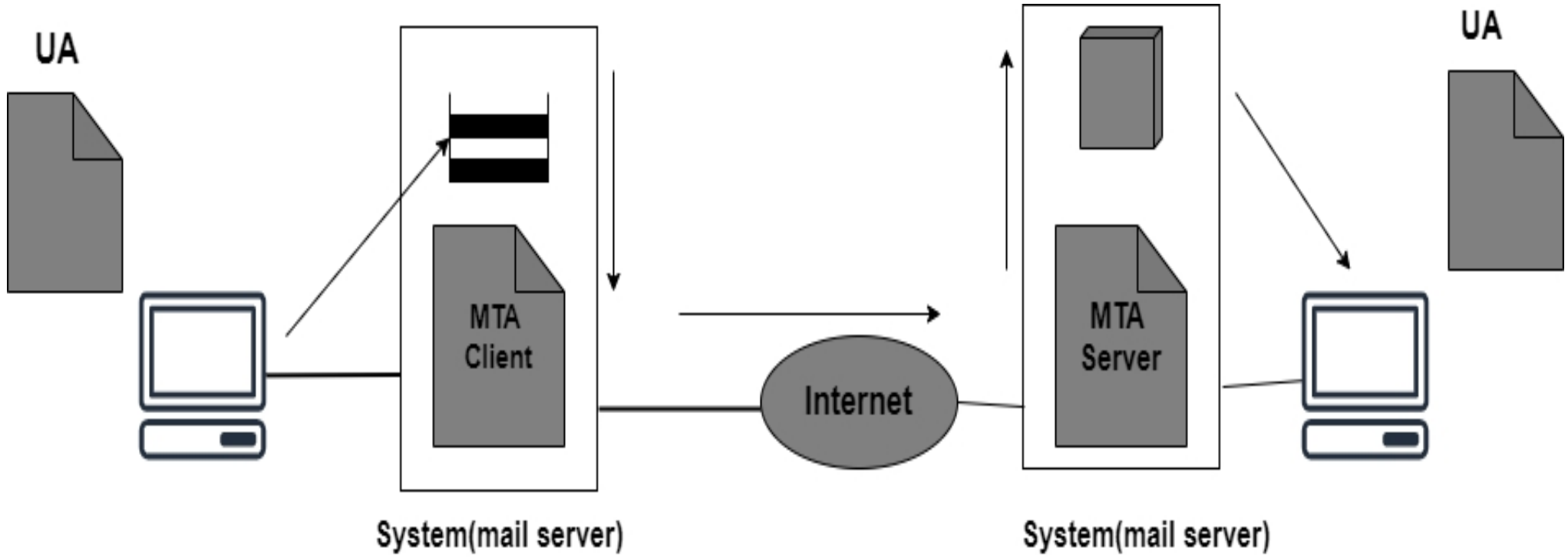
When the sender and the receiver of an E-mail are on the same system, then there is the need for only two user agents.



Second Scenario

In this scenario, the sender and receiver of an e-mail are basically users on the two different systems.

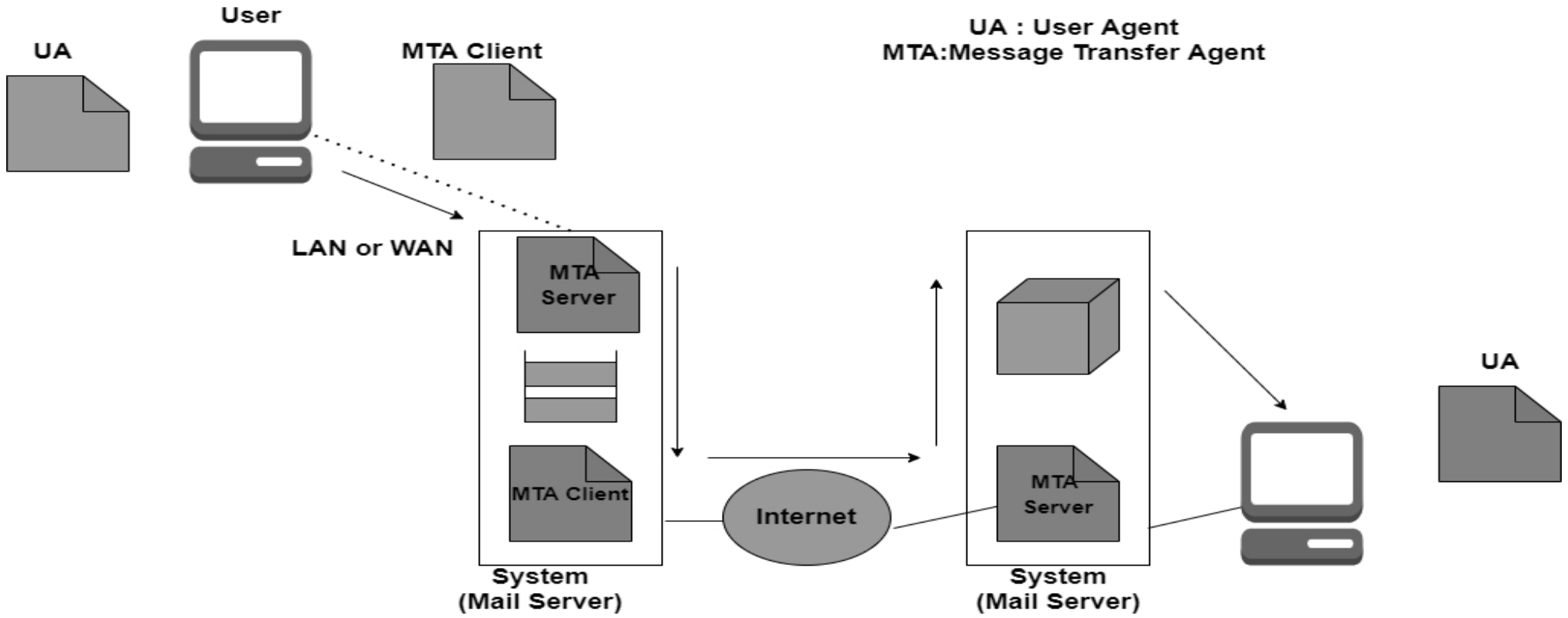
Also, the message needs to send over the Internet. In this case, we need to make use of User Agents and Message transfer agents(MTA).



Third Scenario

In this scenario, the sender is connected to the system via a point-to-point WAN it can be either a dial-up modem or a cable modem. While the receiver is directly connected to the system like it was connected in the second scenario.

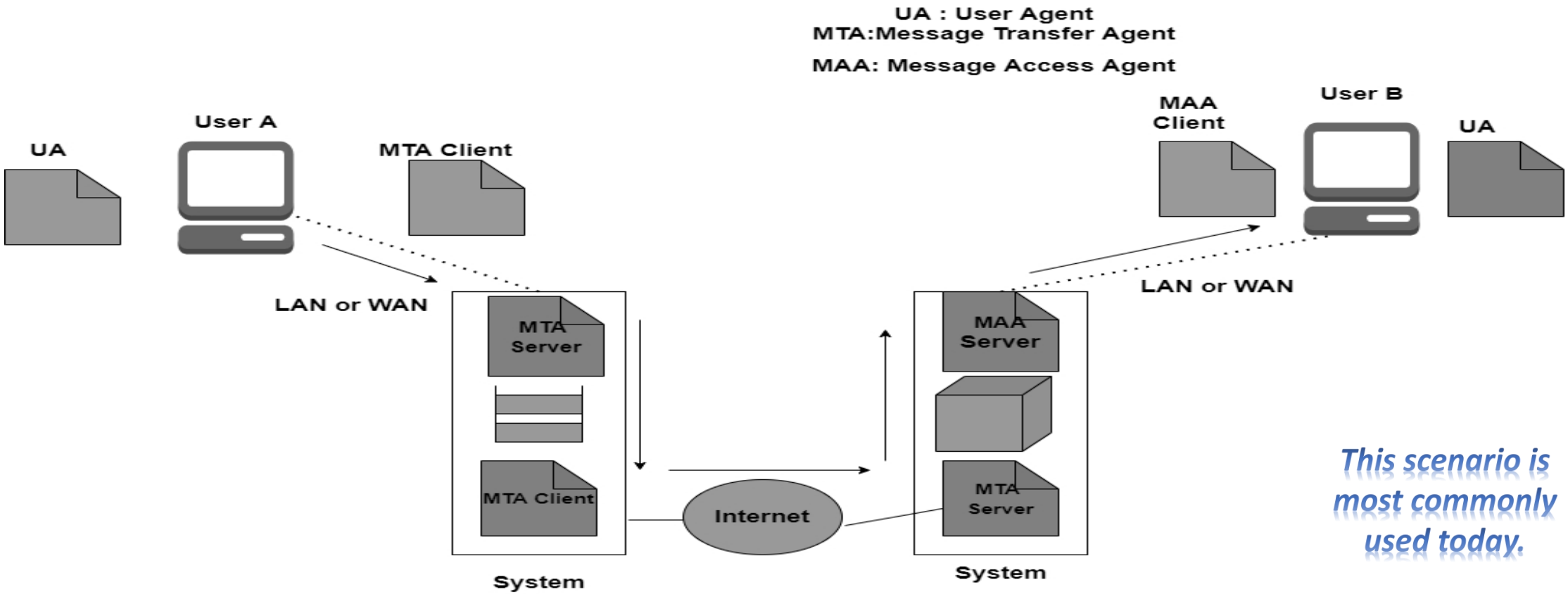
Also in this case sender needs a User agent(UA) in order to prepare the message. After preparing the message the sender sends the message via a pair of MTA through LAN or WAN.



Fourth Scenario

In this scenario, the receiver is also connected to his mail server with the help of WAN or LAN. When the message arrives the receiver needs to retrieve the message; thus there is a need for another set of client/server agents. The recipient makes use of MAA(Message access agent) client in order to retrieve the message.

In this, the client sends the request to the Mail Access agent(MAA) server and then makes a request for the transfer of messages.



This scenario is most commonly used today.

END