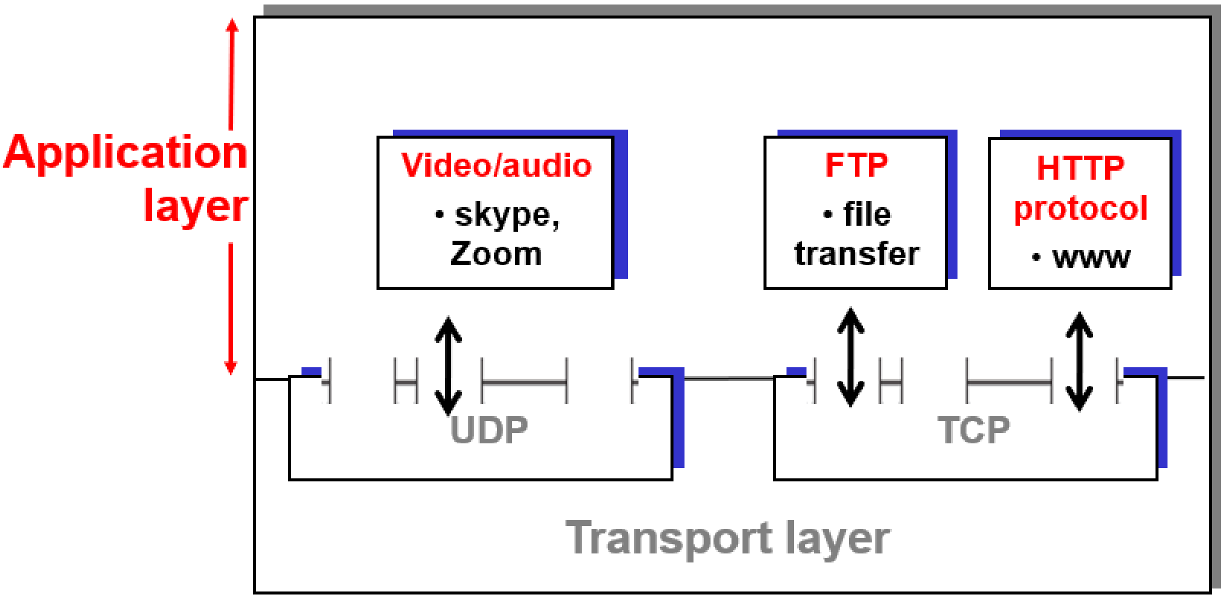
C7 Application Layer

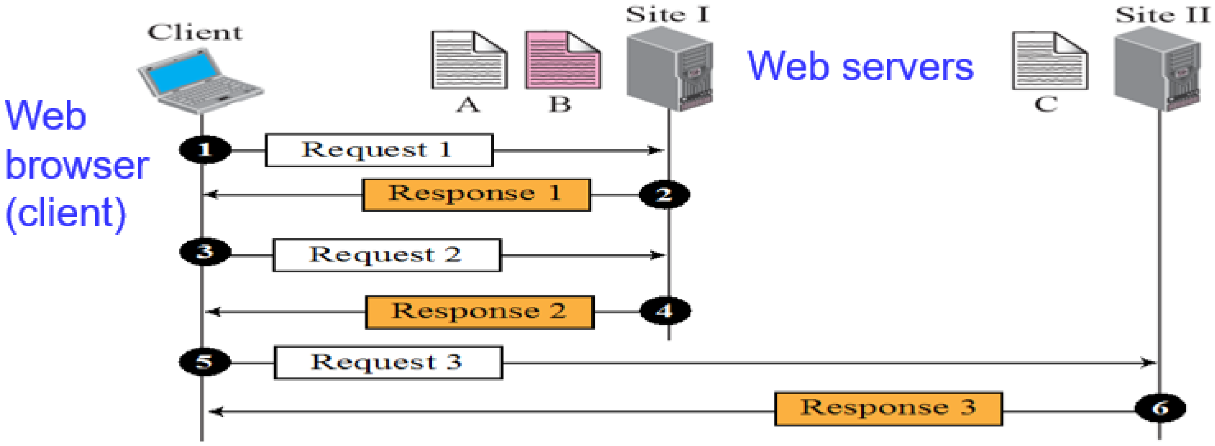
1. Application Layer

- Inside the application layer, there are many protocols / applications, where data is sent from the transport layer to the applications



1.1 World Wide Web (WWW)

- A network application which allows a client to access hypertext file from a server
- **Internet** - a collection of computers and other devices connected by equipment that allows them to communicate with each other
- **Web** - a collection of software and protocols that run as applications of Internet

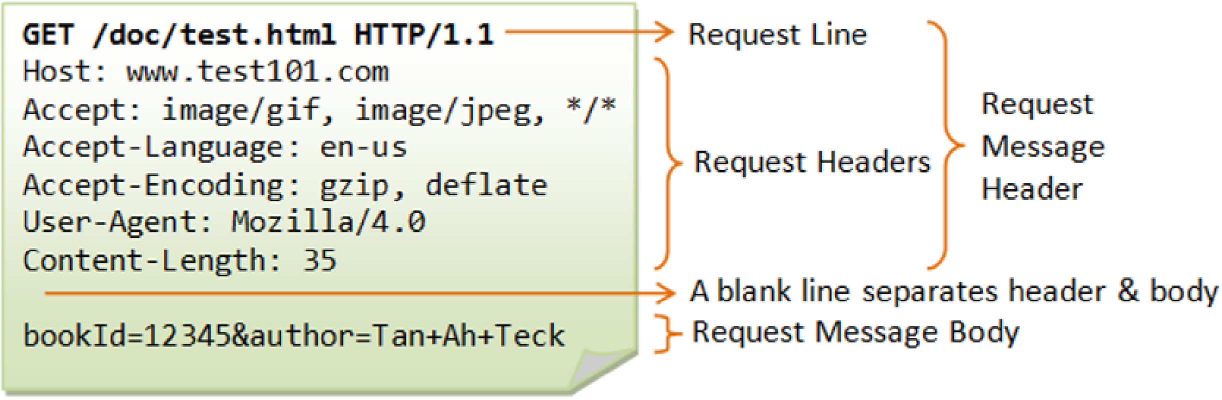


1.2 Hyper Text Transfer Protocol (HTTP)

- Application layer protocol used by WWW and designed to run over **TCP** with server listening at **port 80**, consisting of request / response messages

Requests

- Request forms consist of
  1. `HTTP-method | doc-path-URL | HTTP version`
     - eg: `GET /pub/WWW/file.html HTTP 1.1`
  2. Header fields
  3. *blank line*
  4. Message body
- Common HTTP methods include
  - GET - Fetching a document
  - POST - Executing document using data in body
  - HEAD - Fetch just the header of document
  - PUT - Store new document on server
  - DELETE - Remove document from server
- Common header fields
  - Accept: text/plain - specify browser's preference for type of requested document
    - Accept: *type*/*subtype* - e.g. text/html, image/gif, image/jpeg, text/* (any text)
  - If-Modified-Since: date - send requested document only if it has been modified since specified date
  - Content-Length: number of bytes (for POST requests)
  - Content-Type: text / html (for POST requests)

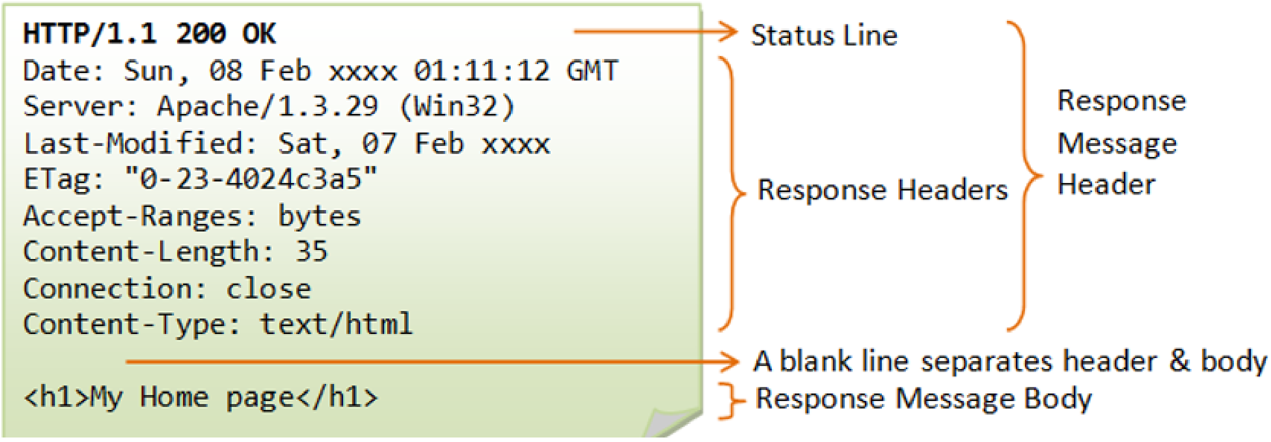


```
GET /doc/test.html HTTP/1.1
Host: www.test101.com
Accept: image/gif, image/jpeg, */*
Accept-Language: en-us
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0
Content-Length: 35

bookId=12345&author=Tan+Ah+Teck
```

Responses

- Response forms consist of
  1. Status line [`HTTP-version | status code | explanation`]
     - eg: `HTTP/1.1 200 OK`
  2. Response header fields
  3. *blank line*
  4. Response body



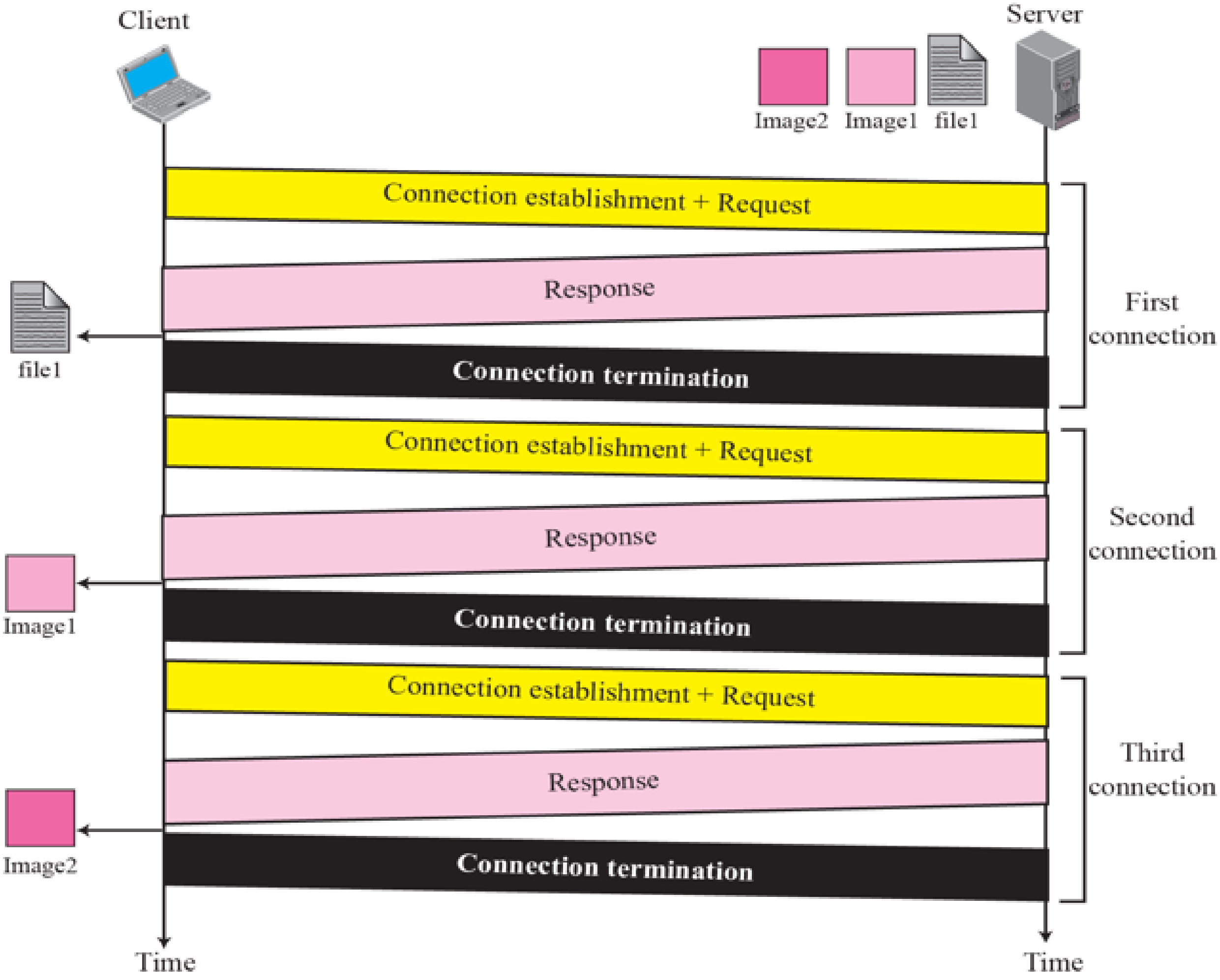
```
HTTP/1.1 200 OK
Date: Sun, 08 Feb xxxx 01:11:12 GMT
Server: Apache/1.3.29 (Win32)
Last-Modified: Sat, 07 Feb xxxx
ETag: "0-23-4024c3a5"
Accept-Ranges: bytes
Content-Length: 35
Connection: close
Content-Type: text/html

<h1>My Home page</h1>
```

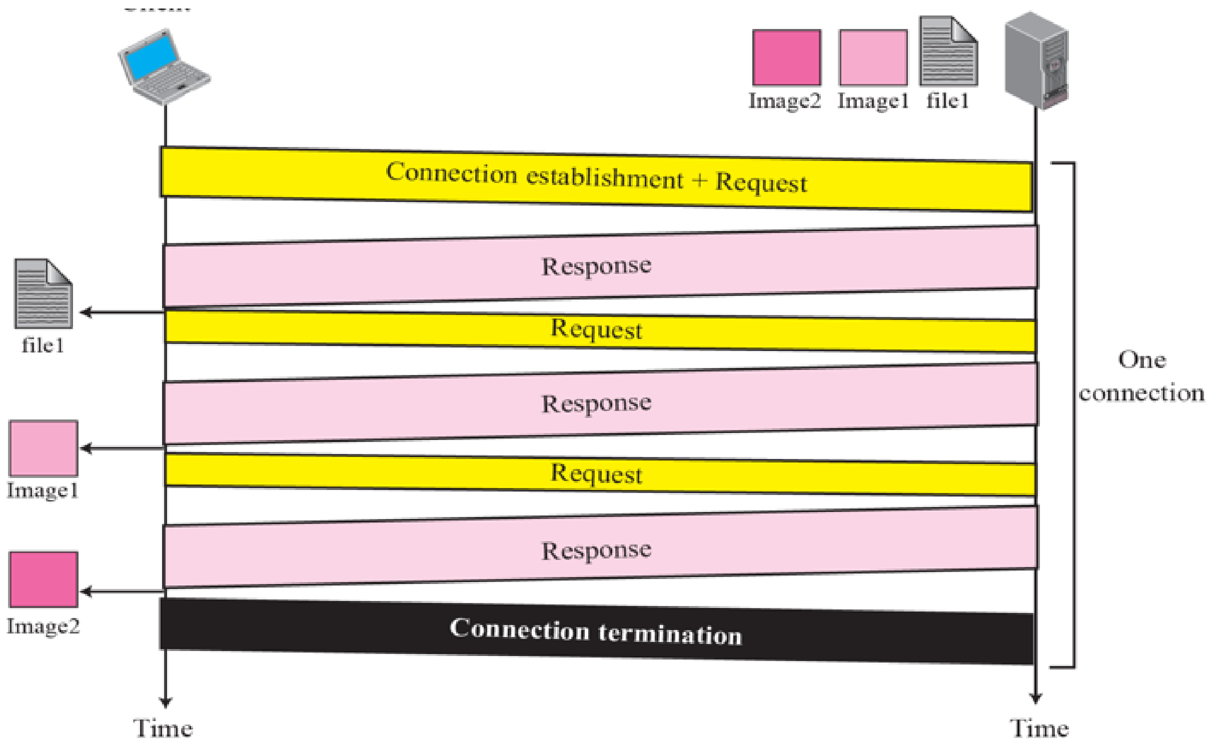
Status Line
Response Headers
Response Message Header
A blank line separates header & body
Response Message Body

Non-persistent / persistent HTTP

- Non-persistent - individual TCP connection / termination for each pair of request / response to access one file; **inefficient**

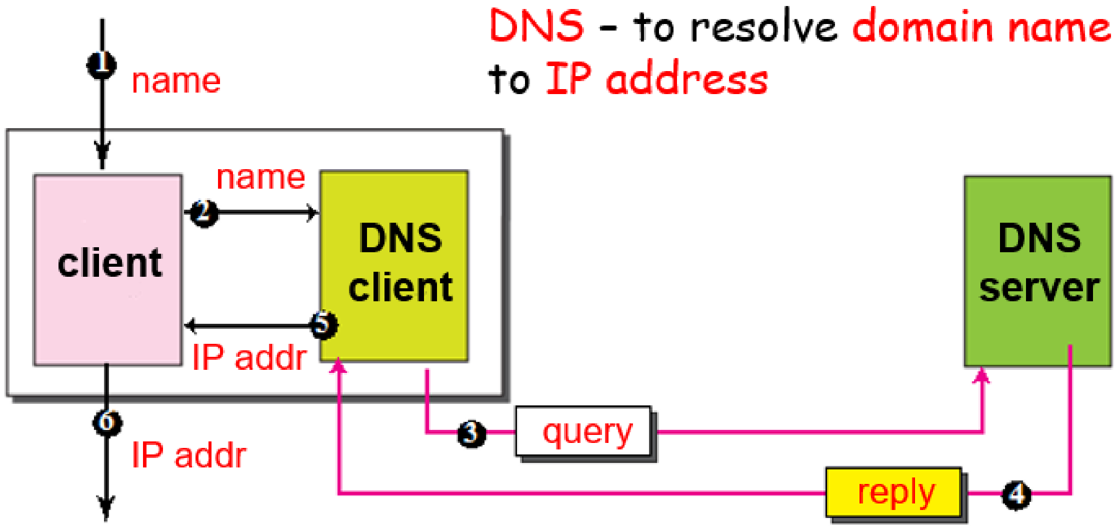


- Persistent - multiple request / response messages within one TCP connection; **efficient for accessing multiple files in the same server**



2. Domain Name System (DNS)

- Resolves domain name to IP address and designed to run **over UDP** with server listening at **port 53**



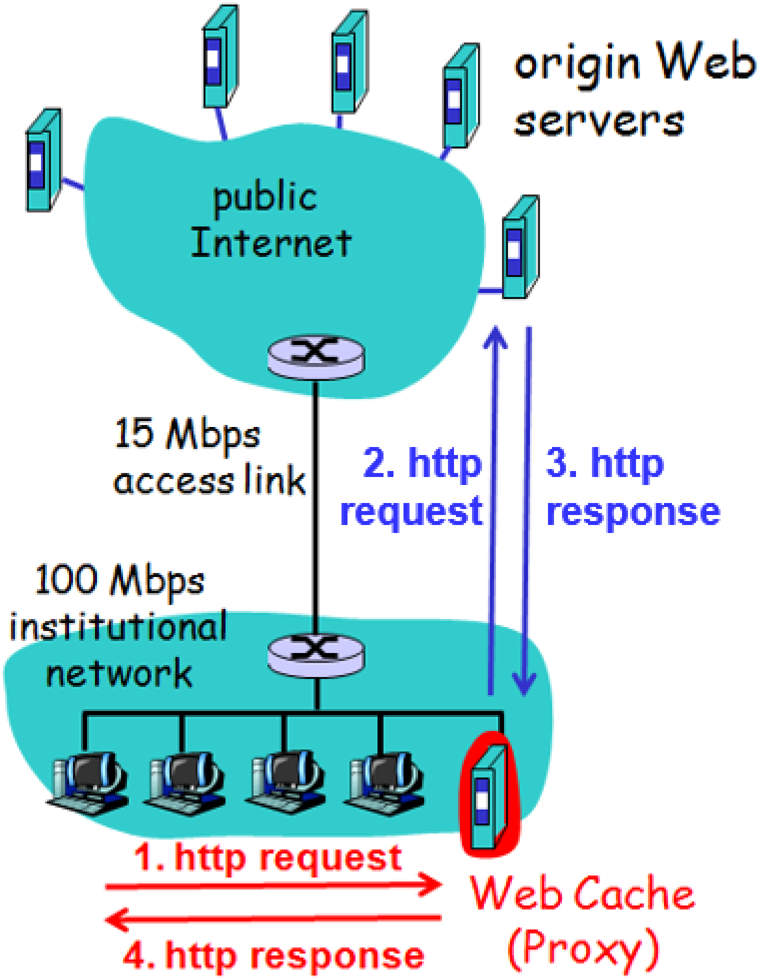
- Domain names are designed to be hierarchical (eg `ece.toronto.edu`, ece is 3rd level, toronto is 2nd level and edu is top level)
  - Top level domains are managed by IANA, and below top level domains, name space is delegated to respective organisations for management, which can further delegate upper lower level name space
- A hierarchy of name servers are set up to provide DNS services with each server being authoritive (responsible) for a zone (either single level or multiple levels) of a DNS name space
  - eg an authoritive server for `virginia.edu` is also authoritive for `xxx.virginia.edu`

How everything works

- To reach an Internet resource, the following need to be specified
  1. Method / Protocol used (application layer)
  2. Host using IP address / domain name (network layer)
  3. Port number (transport layer)
  4. Path and document name (application layer)
- All of which are combined into a single string called URL

3. Web Proxy (Cache)

- Improves performance by caching, reducing traffic load on costly access link and at the same time monitor / filter contents
  1. Client requests to proxy
  2. (if content is not available) proxy requests to origin servers
  3. Origin server responds to proxy
  4. Proxy responds to client



- Can be implemented transparently without the knowledge of users / servers
  - Organisations / ISPs configure routers to intercept all traffic and re-direct them to its web-proxy which masquerades (acts as) as the destination server

- **Content Delivery / Distribution Network** can be used as a content provider if performance is important