Module 2: Application Layer

(Lecture-1)

Dr. Nirnay Ghosh

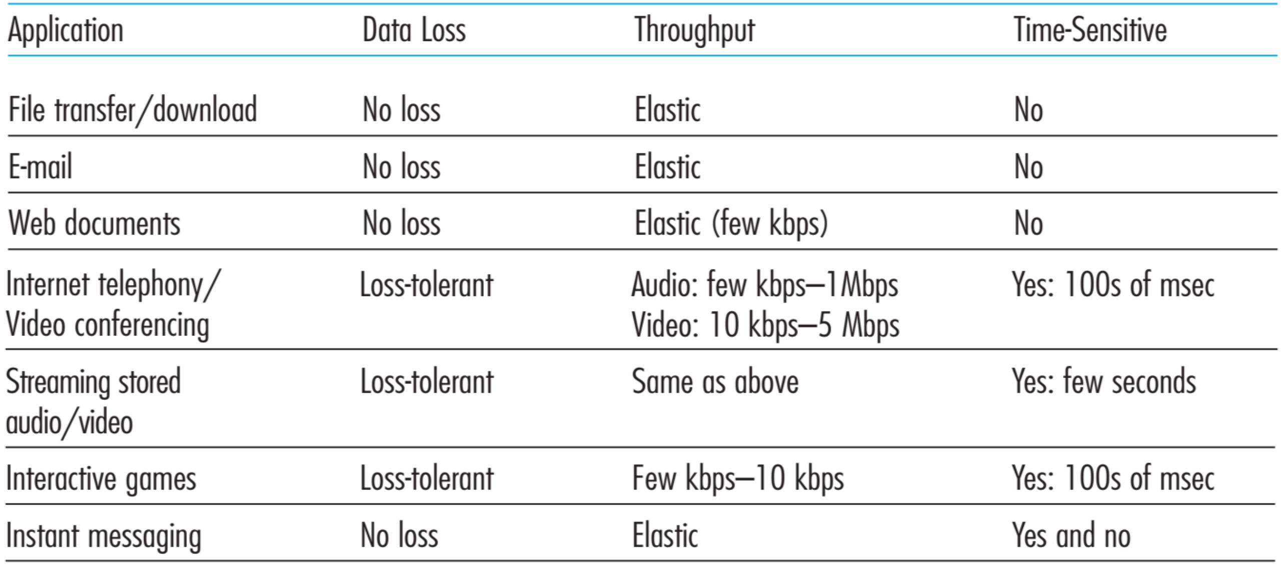
Assistant Professor

Department of Computer Science & Technology

IIEST, Shibpur

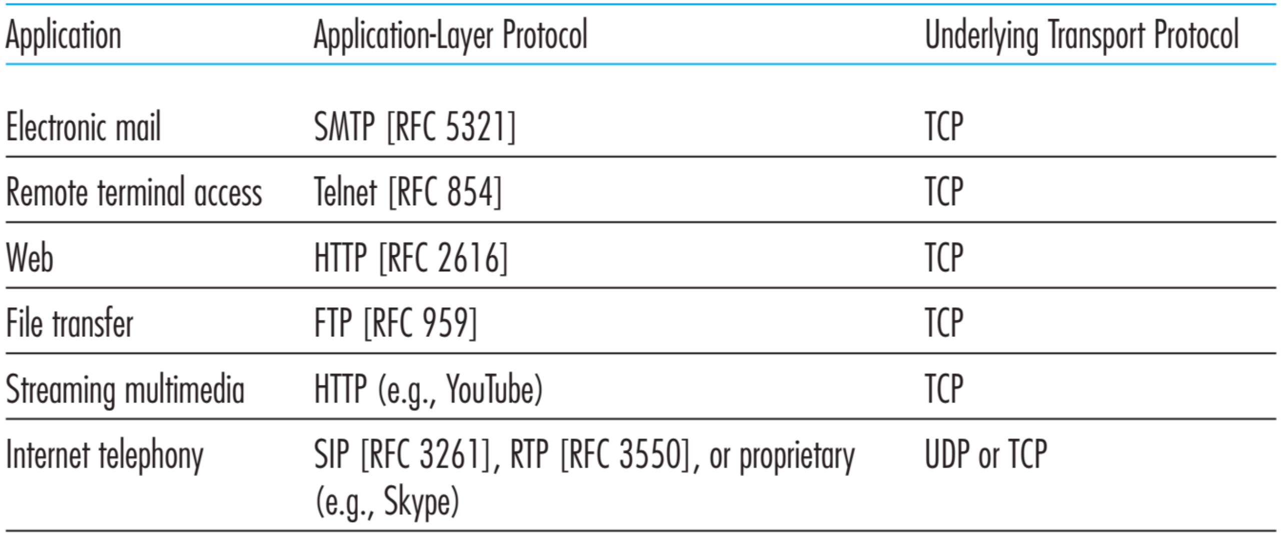
Application Layer Protocols

- Defines how an application's processes, running on different end systems, pass messages to each other
  - Types of messages exchanged (e.g., request message and response message)
  - Syntax of various message types (e.g., fields in the message and how they are separated)
  - Semantics of the fields (i.e. meaning of the fields)
  - Rules for determining when and how a process sends messages and responds to messages
- Can be both public and proprietary
  - Public: HTTP (Hyper Text Transfer Protocol) - Web's application-layer protocol
  - Proprietary: Skype's application-layer protocols
- Forms only one piece of a network application
  - Example: (i) Web application: standard for document formats (HTML), Web browsers, Web servers, and an application layer-protocol (HTTP); (ii) Internet's e-mail application: mail server, mail client, application-layer protocol (e.g., SMTP)



Application	Data Loss	Throughput	Time-Sensitive
File transfer/download	No loss	Elastic	No
E-mail	No loss	Elastic	No
Web documents	No loss	Elastic (few kbps)	No
Internet telephony/ Video conferencing	Loss-tolerant	Audio: few kbps—1Mbps Video: 10 kbps—5 Mbps	Yes: 100s of msec
Streaming stored audio/video	Loss-tolerant	Same as above	Yes: few seconds
Interactive games	Loss-tolerant	Few kbps—10 kbps	Yes: 100s of msec
Instant messaging	No loss	Elastic	Yes and no

Requirements of Selected Network Applications



Application	Application-Layer Protocol	Underlying Transport Protocol
Electronic mail	SMTP [RFC 5321]	TCP
Remote terminal access	Telnet [RFC 854]	TCP
Web	HTTP [RFC 2616]	TCP
File transfer	FTP [RFC 959]	TCP
Streaming multimedia	HTTP (e.g., YouTube)	TCP
Internet telephony	SIP [RFC 3261], RTP [RFC 3550], or proprietary (e.g., Skype)	UDP or TCP

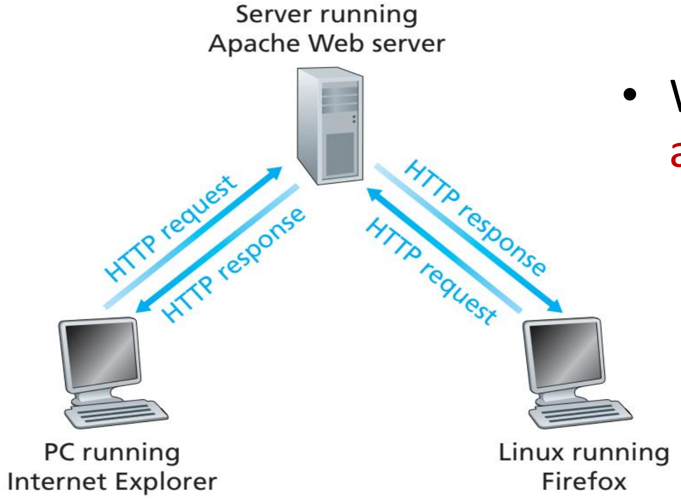
Popular Internet Applications with Application-layer and Transport Layers Protocols

The Web & HTTP

- Some features of the Web
  - On-demand: users receive what they want, when they want
  - Easy for any individual to make information available over the Web – everyone can become a publisher at extremely low cost
  - Hyperlinks and search engines help in navigating through an ocean of Web sites
  - Forms, JavaScript, Java applets, and many other technologies enable us to interact with pages and sites
- Web page (also called a document) consists of:
  - Referenced objects (e.g., files such as JPEG image, a Java applet, or a video clip)
  - Base HTML file: references the objects in the page with the object's URL (Universal Resource Locator)
- URL – two components: hostname of the server that houses the object; object's path name

```
http://www.someSchool.edu/someDepartment/picture.gif
```

Hostname **Object's Pathname**

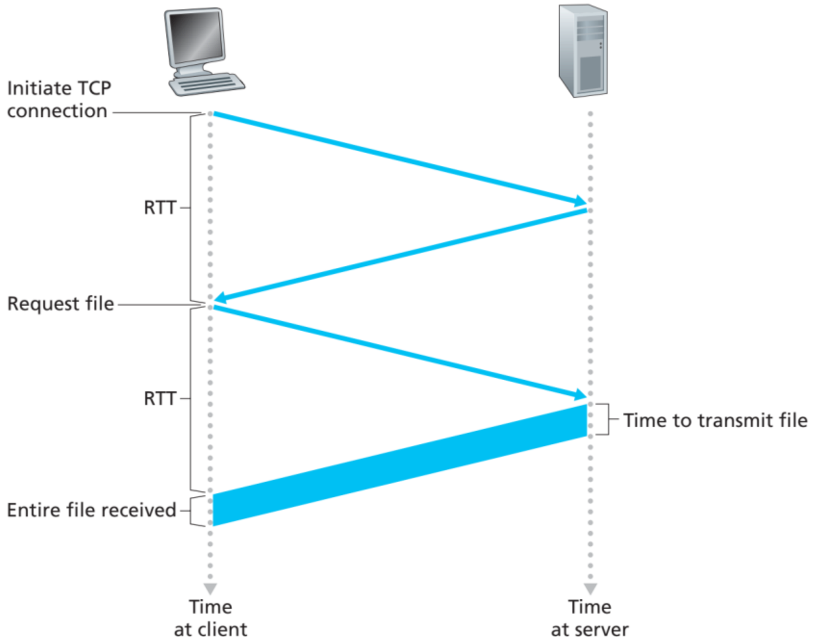


Web's Client-Server Architecture

- Web uses client server architecture
  - Web browsers (e.g., IE, Firefox, Chrome): implement the client side of HTTP
  - Web servers (e.g., Apache, Microsoft IIS): implement the server side of the HTTP; always ON with fixed IP address; can serve potentially millions of different clients
- Overview of HTTP
  - Web's application-layer protocol, uses the underlying transport protocol (TCP)
  - Implemented in two programs – client and server
  - HTTP defines the structure of the message and how Web clients/browsers request Web pages from Web servers and how servers transfer Web pages to clients.
  - Stateless protocol – maintains no information about the clients
  - Supports both persistent (default) and non-persistent connections

HTTP: Persistent & Non-persistent Connections

- Non-persistent connection: each request/response pair are sent over a separate TCP connection (HTTP 1.0)
  - TCP connection transports exactly one request message and one response message
  - If a user requests for a Web page with *n* objects, then the same number of TCP connections are generated (either serially or parallelly)
  - Browsers can control the degree of parallelism – by default 5 to 10 parallel TCP connections can be opened, each one handling a request-response pair
  - Total response time to receive base HTML file/object from the server is:
    - 2 * RTT (Round-Trip Time) + base HTML file/object transfer time at the server
  - Connection is closed after the server sends the object (i.e., does not persist for other objects)
  - Overhead:
    - Repeated allocation of TCP buffers and variables in both client and server
    - Each object suffers a minimum delay of 2*RTT

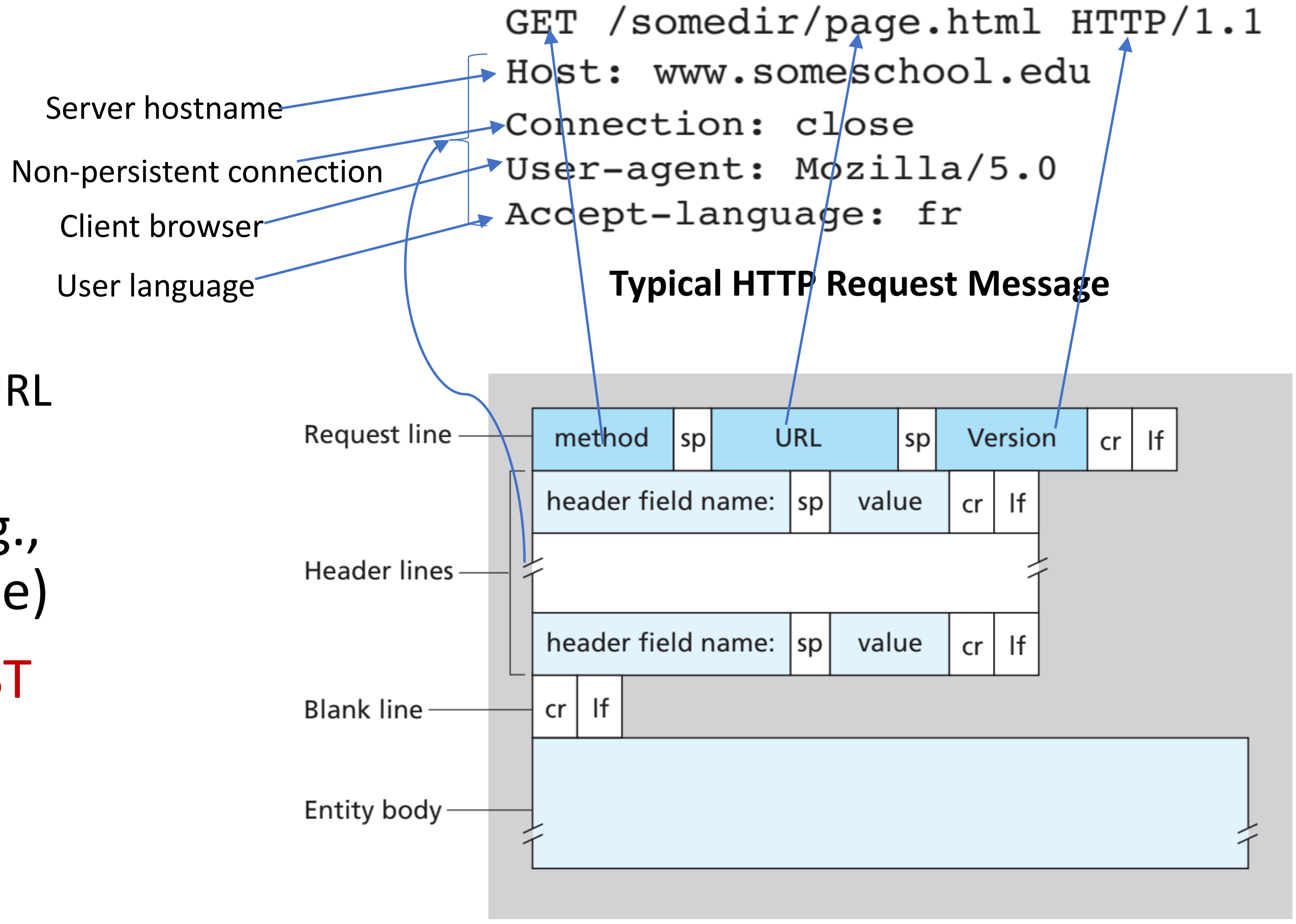


Calculation for the Time needed to Request and Receive an HTML file

- Persistent connection: all request/response pair are sent over the same TCP connection (HTTP 1.1)
  - Default mode for HTTP 1.1 (persist connections with pipelining)
  - Server leaves the TCP connection open after sending the response
  - Subsequent requests and responses between same client-server pair can be sent over the same connection
  - Requests for objects can be made back-to-back without waiting for the pending responses (pipelining); Server sends the objects back-to-back
  - Server closes a connection when it isn't used for a configurable timeout interval

HTTP: Request Format

- Request methods: GET, POST, PUT, and DELETE
- Majority of requests use GET method
  - Used when the browser requests an object, with the requested object identified in the URL field *(/somedir/page.html)*
- POST: used if the user fills out a form (e.g., providing search words to a search engine)
- Entity body: empty for GET; used for POST
- HTML often uses the GET method and include the inputted form data in the requested URL
- PUT: web publishing tool
  - Used to upload an object to a specific path (directory) on a specific Web server
- DELETE: allows an user or an application to delete an object on the Web server



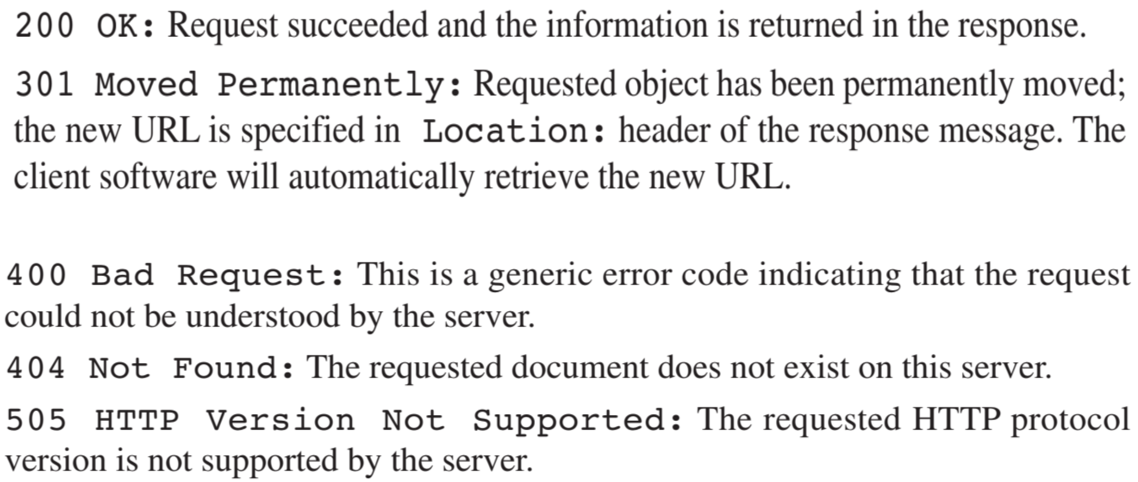
```
GET /somedir/page.html HTTP/1.1
Host: www.someschool.edu
Connection: close
User-agent: Mozilla/5.0
Accept-language: fr
```

Typical HTTP Request Message

General Format of an HTTP Request Message

HTTP Response Format

- Consists of: status line, header lines, entity body
- Status code & associated phrase: result of the request
    - Some of the common status codes and associated phrases are as follows:



`200 OK`: Request succeeded and the information is returned in the response.

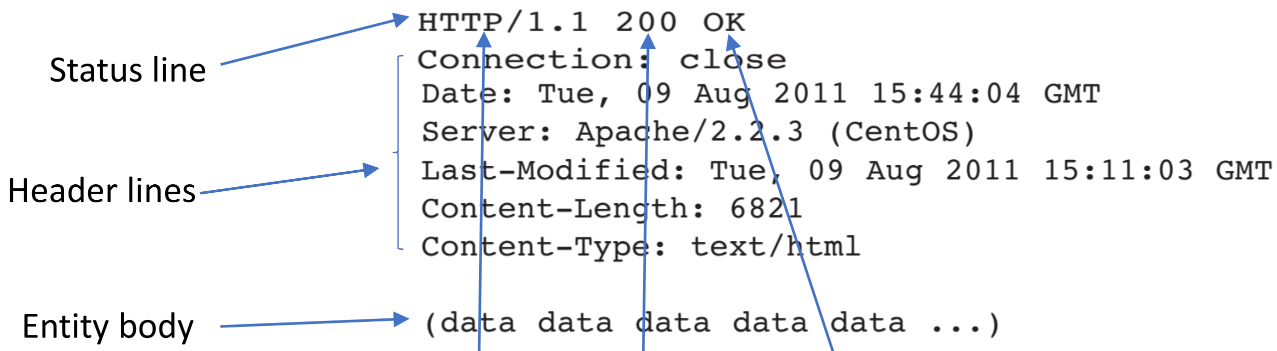
`301 Moved Permanently`: Requested object has been permanently moved; the new URL is specified in `Location:` header of the response message. The client software will automatically retrieve the new URL.

`400 Bad Request`: This is a generic error code indicating that the request could not be understood by the server.

`404 Not Found`: The requested document does not exist on this server.

`505 HTTP Version Not Supported`: The requested HTTP protocol version is not supported by the server.

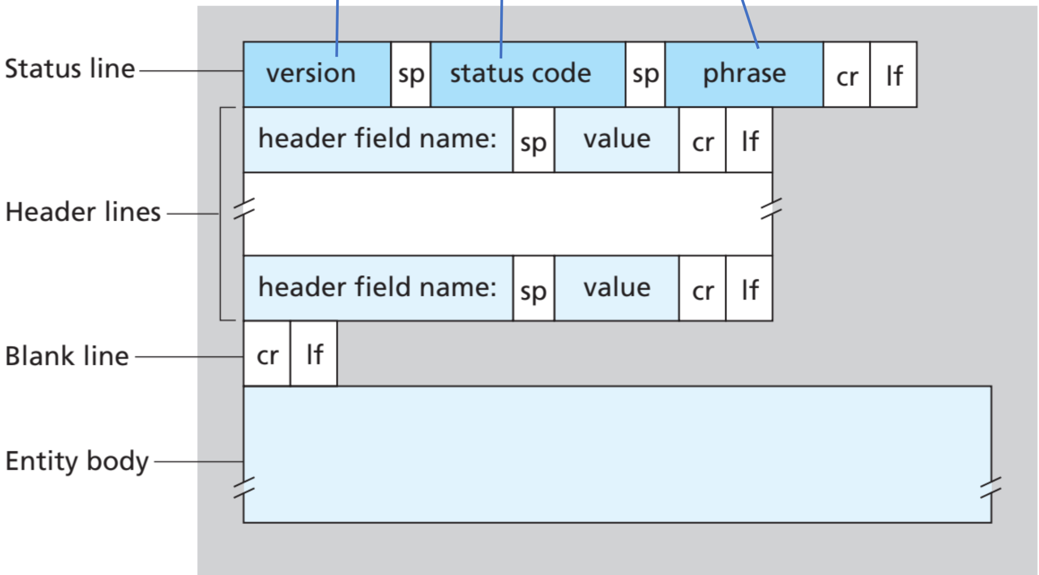
- Header lines: fields are self-explanatory
- Entity body: contains the requested object itself
- HEAD (response message): server responds with an HTTP message but leaves out the requested object
    - Useful for retrieving meta information written in the response headers



```
HTTP/1.1 200 OK
Connection: close
Date: Tue, 09 Aug 2011 15:44:04 GMT
Server: Apache/2.2.3 (CentOS)
Last-Modified: Tue, 09 Aug 2011 15:11:03 GMT
Content-Length: 6821
Content-Type: text/html

(data data data data data ...)
```

Typical HTTP Request Message



General Format of an HTTP Response Message