Presentation on

Web Technology

Topic: HTTP server and HTTP protocol

Course Code: CSE-502

Presented by:

Md. Rakib Hossain, BSSE-0516
A. H. M. Azimul Haque, BSSE-0519
Md. Mahbub Islam, BSSE-0510
Md. Rashedul Islam, BSSE-0510

Submitted to:

Mr. Amit Seal Ami
Lecturer,
Institute of Information Technology
University of Dhaka

What is the WWW?

- Is a system of interlinked hypertext documents
- Accessed via the Internet with a web browser
- Client/Server data transfer protocol
- Communication via application level protocol
- Run on standard networking infrastructure

Internet V/s WWW

Internet

- ✓ The Internet is a global system of interconnected computer networks.
- ✓ Its access is provided by ISPs.
- ✓ It runs applications like www, ftp, html etc

World Wide Web

- ✓ Web is collection of text documents and other resources, linked by hyperlinks and URLs
- ✓ Usually accessed by web browsers
- ✓ Its an application running on Internet

What is the Web Pages?

- Web page consists of objects
- Object can be HTML file, JPEG image, Java applet, audio file,…
- Web page consists of base HTML-file which includes several referenced objects
- Each object is addressable by a URL
- Example URL:

www.someschool.edu/someDept/pic.gif

host name | path name

What is the URL?

<scheme> : //<host> :<port> /<path> ;<parameters> ?<query> #<fragment>

Here

scheme
: The protocol you are using

host
: Host name or ip number

port
: TCP port number that protocol server is using

path
: Path and filename reference of object on server

parameters
: Any specific parameters that object needs

query
: Query string for a CGI program

fragment
: Reference to a subset of an object

HTTP Overview

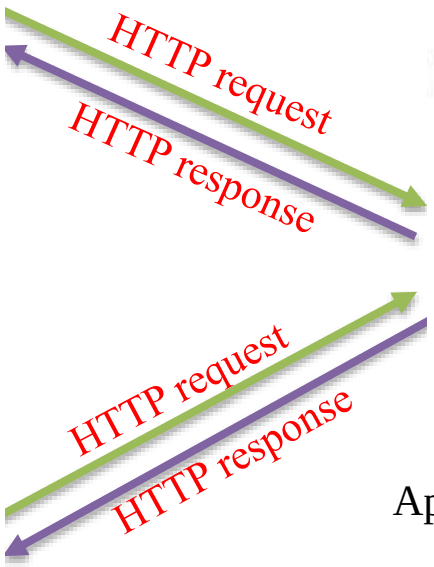
HTTP: hypertext transfer protocol

- Web's application layer protocol
- client/server model
  - ***client:*** browser that requests, receives, "displays" Web objects
  - ***server:*** Web server sends objects in response to requests
- HTTP 1.0: RFC 1945
- HTTP 1.1: RFC 2068

PC running Explorer

Mac running Navigator

Server running Apache Web server



HTTP Overview Con…

Uses TCP:

- client initiates TCP connection (creates socket) to server, port 80
- server accepts TCP connection from client
- HTTP messages (application-layer protocol messages) exchanged between browser (HTTP client) and Web server (HTTP server)
- HTTP – not worry about lost data; taken care of by TCP
- TCP connection closed

HTTP is "stateless"

- server maintains no information about past client requests

Protocols that maintain "state" are complex!

- past history (state) must be maintained
- if server/client crashes, their views of "state" may be inconsistent, must be reconciled

HTTP connections

Non-persistent HTTP

- At most one object is sent over a TCP connection.

- HTTP/1.0 uses nonpersistent HTTP

HTTP connections

Persistent HTTP

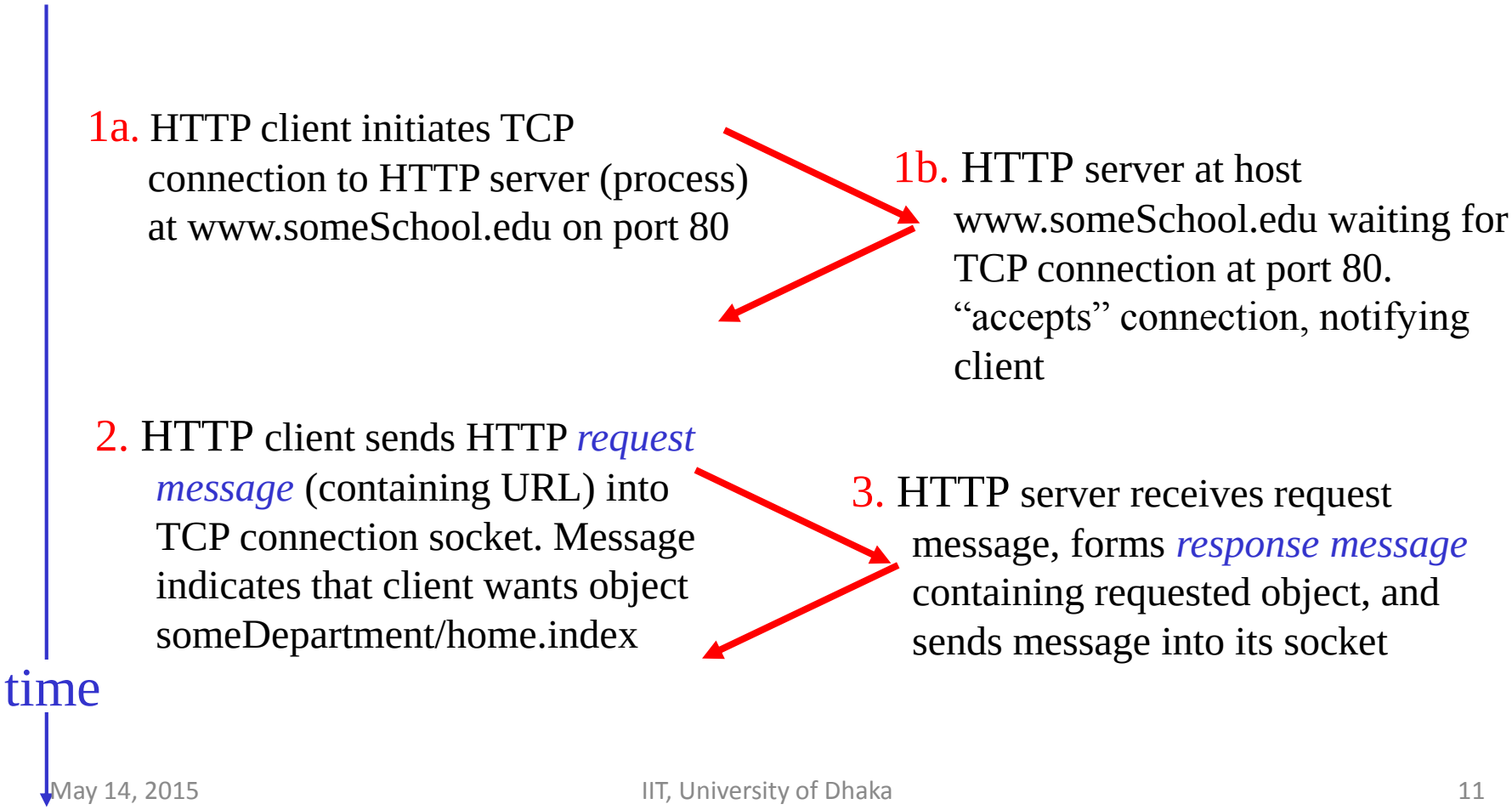
- Multiple objects can be sent over single TCP connection between client and server.

- HTTP/1.1 uses persistent connections in default mode

Non-persistent HTTP

Suppose user enters URL
www.someSchool.edu/someDepartment/home.index

(contains text, references to 10 jpeg images)



Non-persistent HTTP

4. HTTP server closes TCP connection. TCP waits and terminates the connection when it knows that the message is received by the client

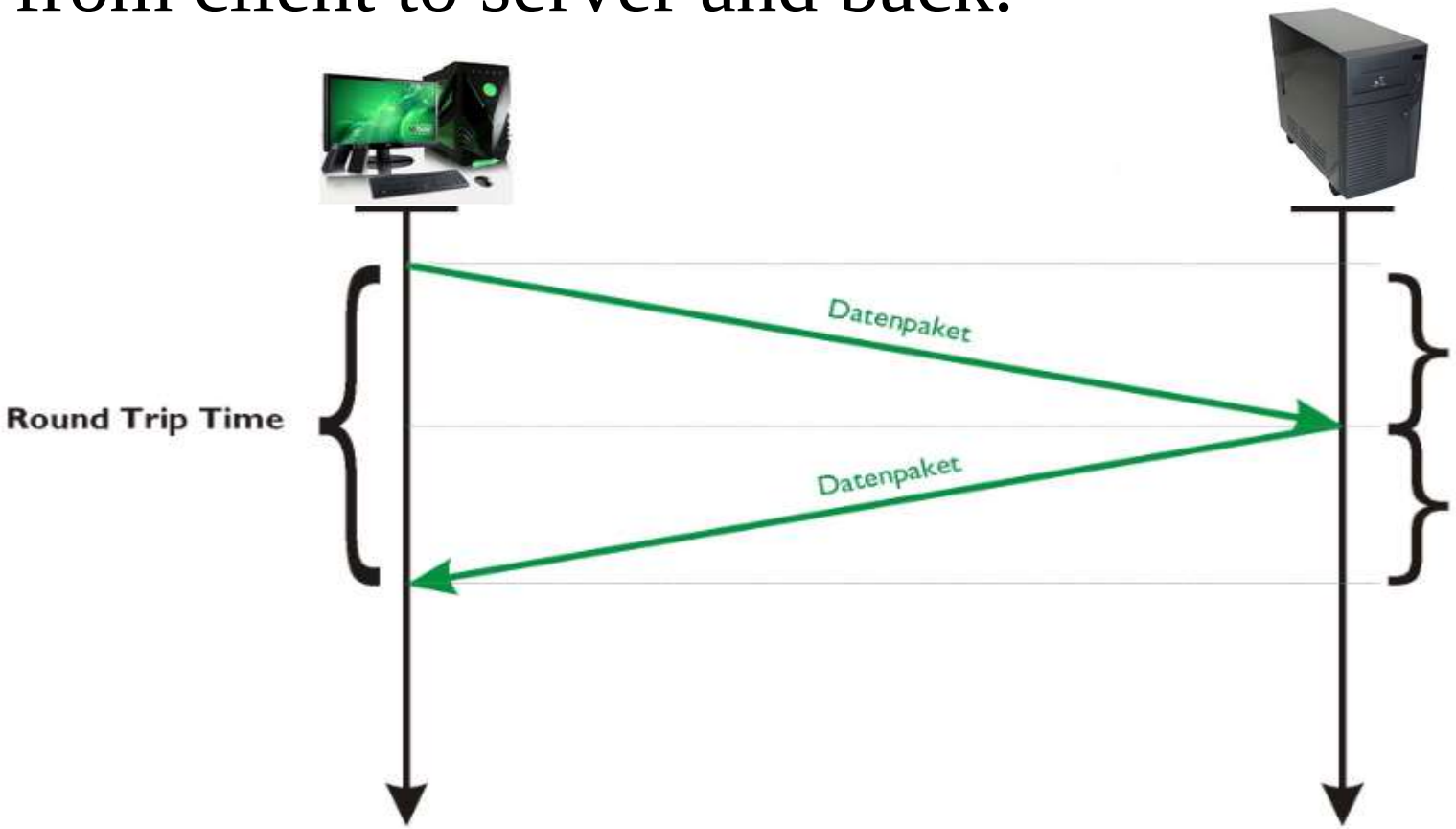
5. HTTP client receives response message containing html file, displays html. Parsing html file, finds 10 referenced jpeg objects

6. Steps 1-5 repeated for each of 10 jpeg objects

Serial vs. parallel TCP connections

HTTP: Response time

RTT: time to send a small packet to travel from client to server and back.

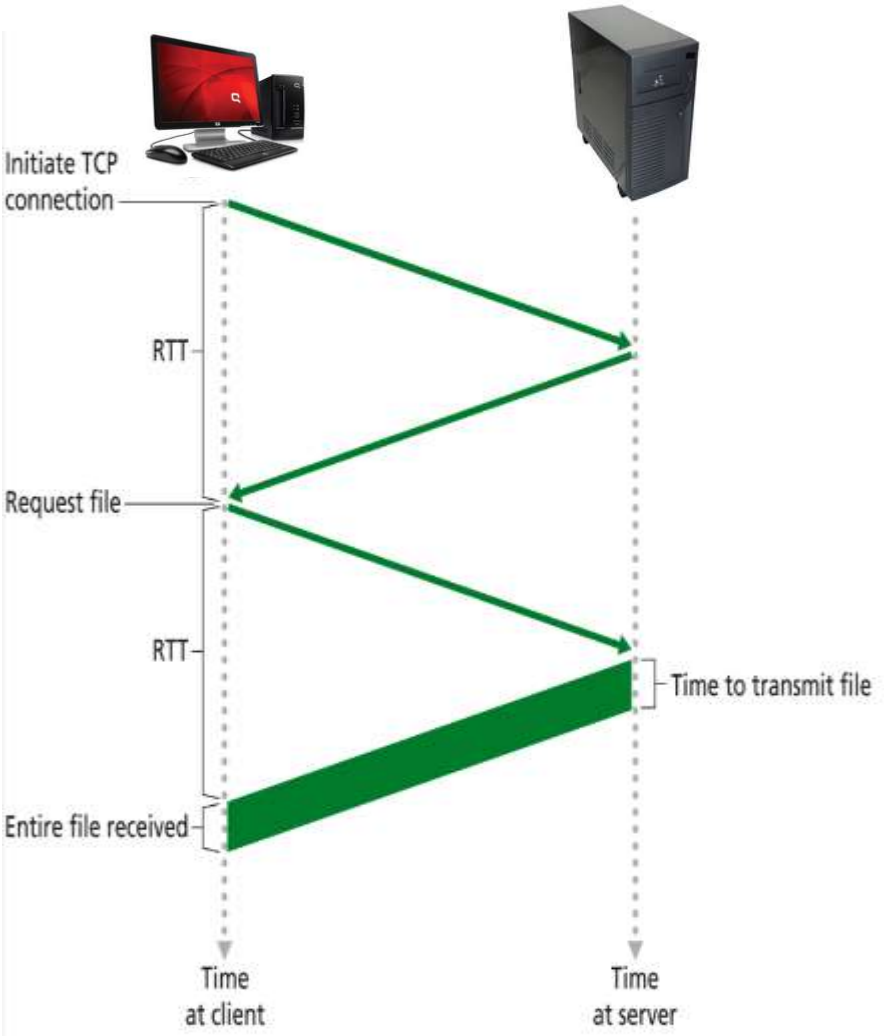


Non-Persistent HTTP: Response time

Response time:

- one RTT to initiate TCP connection
- one RTT for HTTP request and first few bytes of HTTP response to return
- file transmission time

total = 2RTT+transmit time



Non-Persistent HTTP

Non-persistent HTTP issues:

- requires 2 RTTs per object
- OS overhead for *each* TCP connection
- browsers often open parallel TCP connections to fetch referenced objects
- Serious burden on Web server

Persistent HTTP

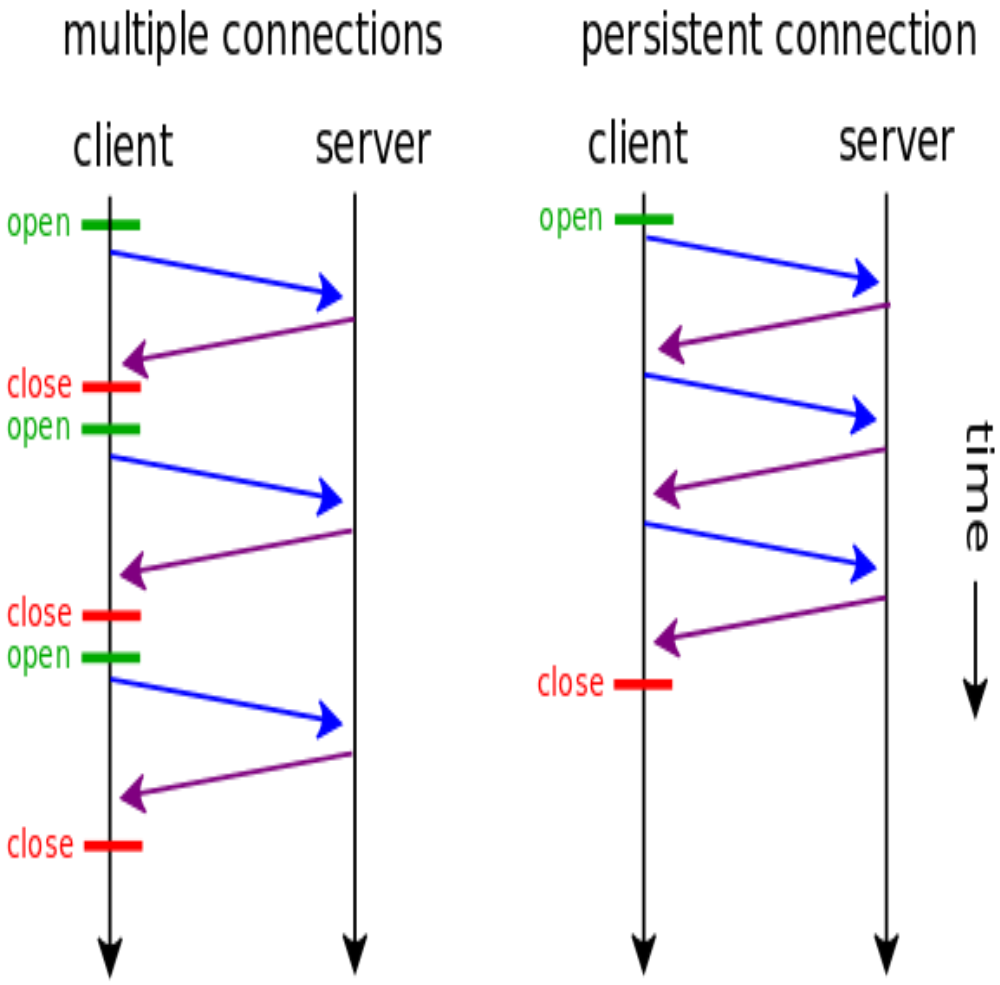
Persistent HTTP

- server leaves connection open after sending response
- subsequent HTTP

messages between same client/server sent over open connection

- Closes connection after

TIMEOUT INTERVAL!!!



Persistent Connection and Pipelining

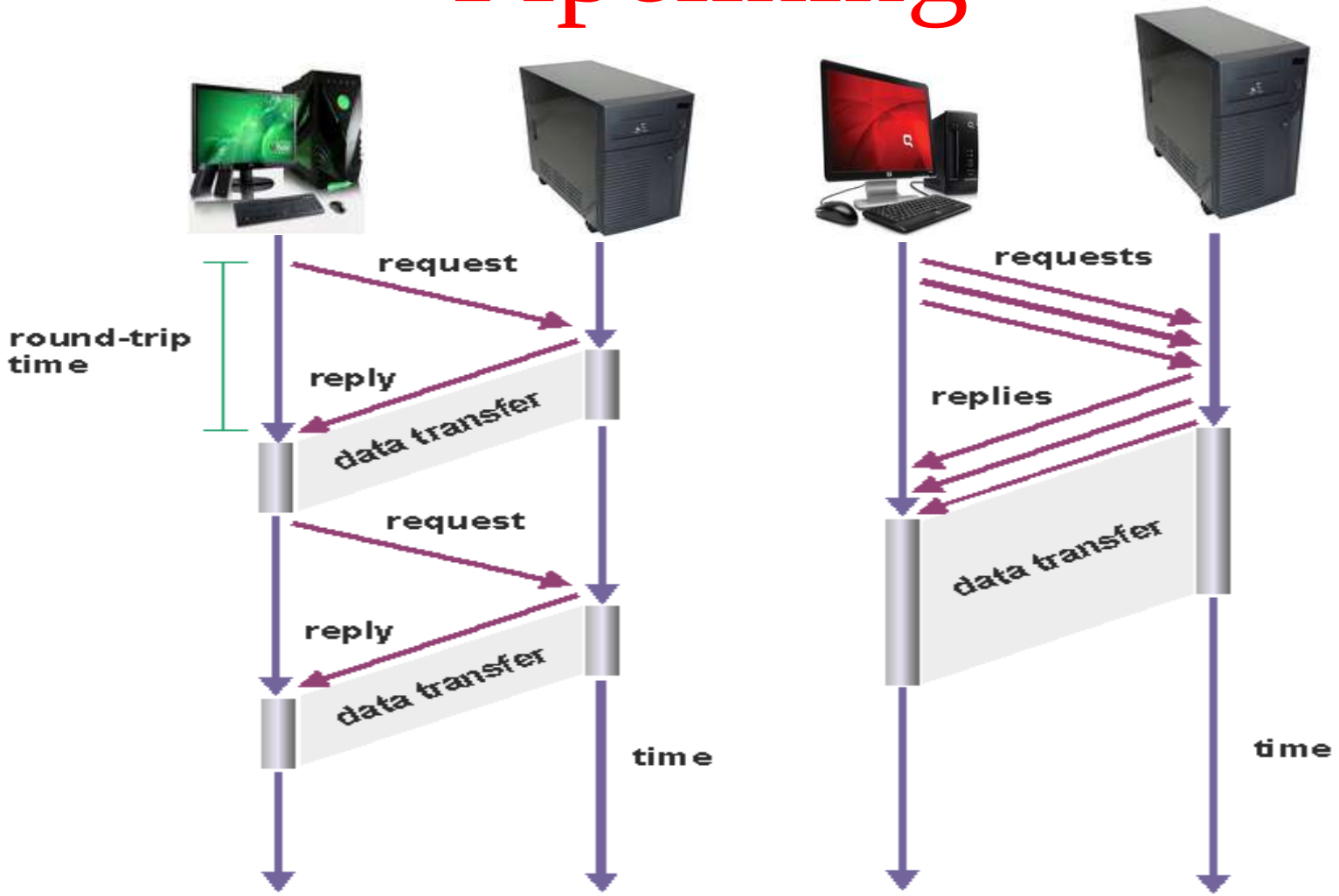
Persistent *without* pipelining:

- client issues new request only when previous response has been received
- one RTT for each referenced object

Persistent *with* pipelining:

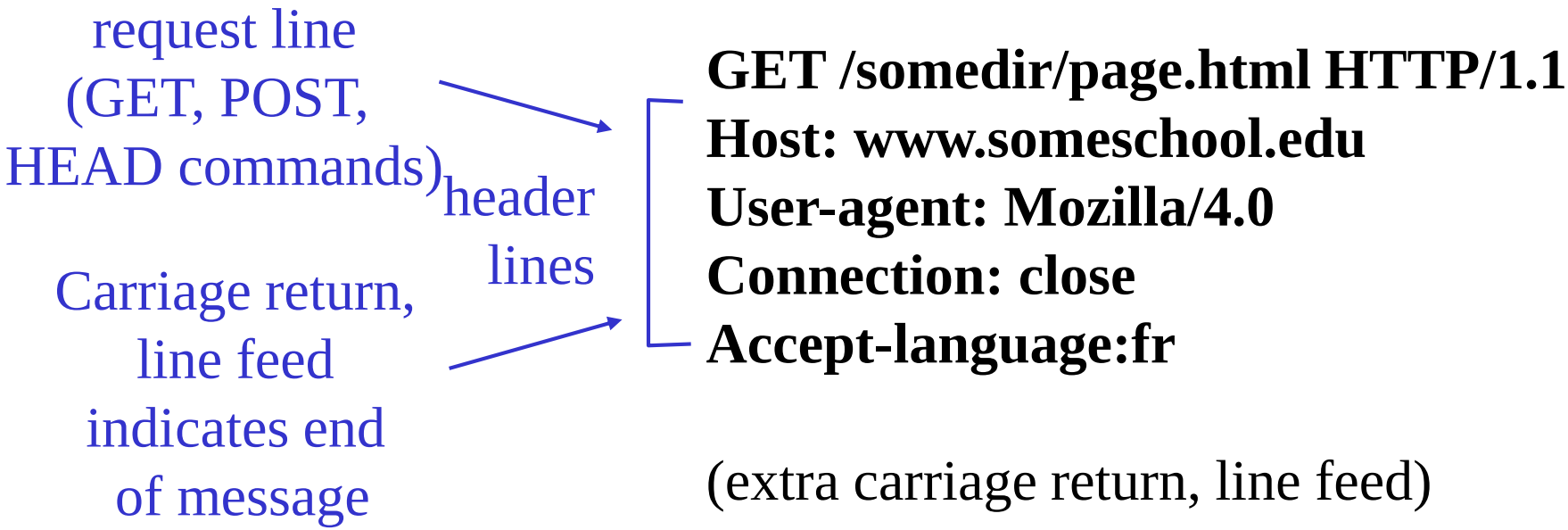
- default in HTTP/1.1
- client sends requests as soon as it encounters a referenced object
- as little as one RTT for all the referenced objects

Persistent Connection and Pipelining

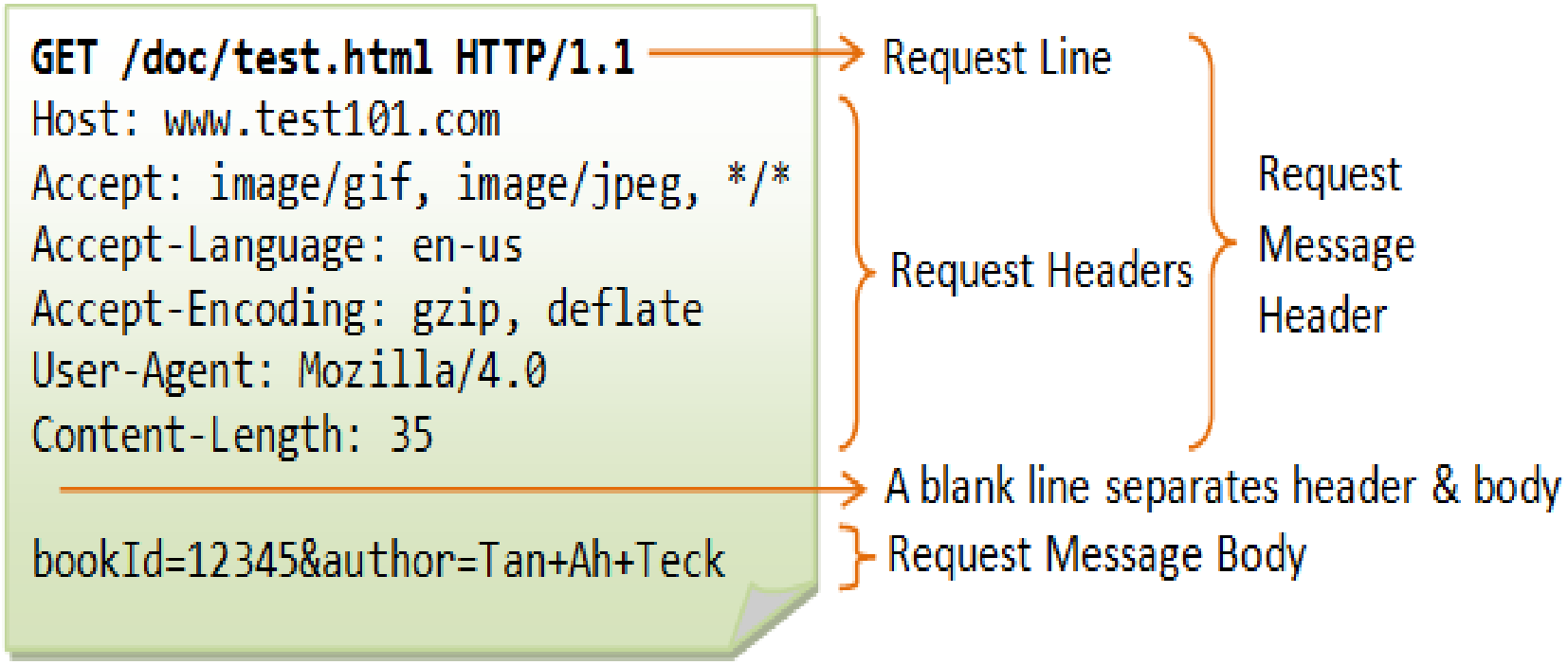


HTTP request message

- two types of HTTP messages: *request, response*
- HTTP request message:
  - ASCII (human-readable format)



HTTP request message Con…



HTTP request message: general format

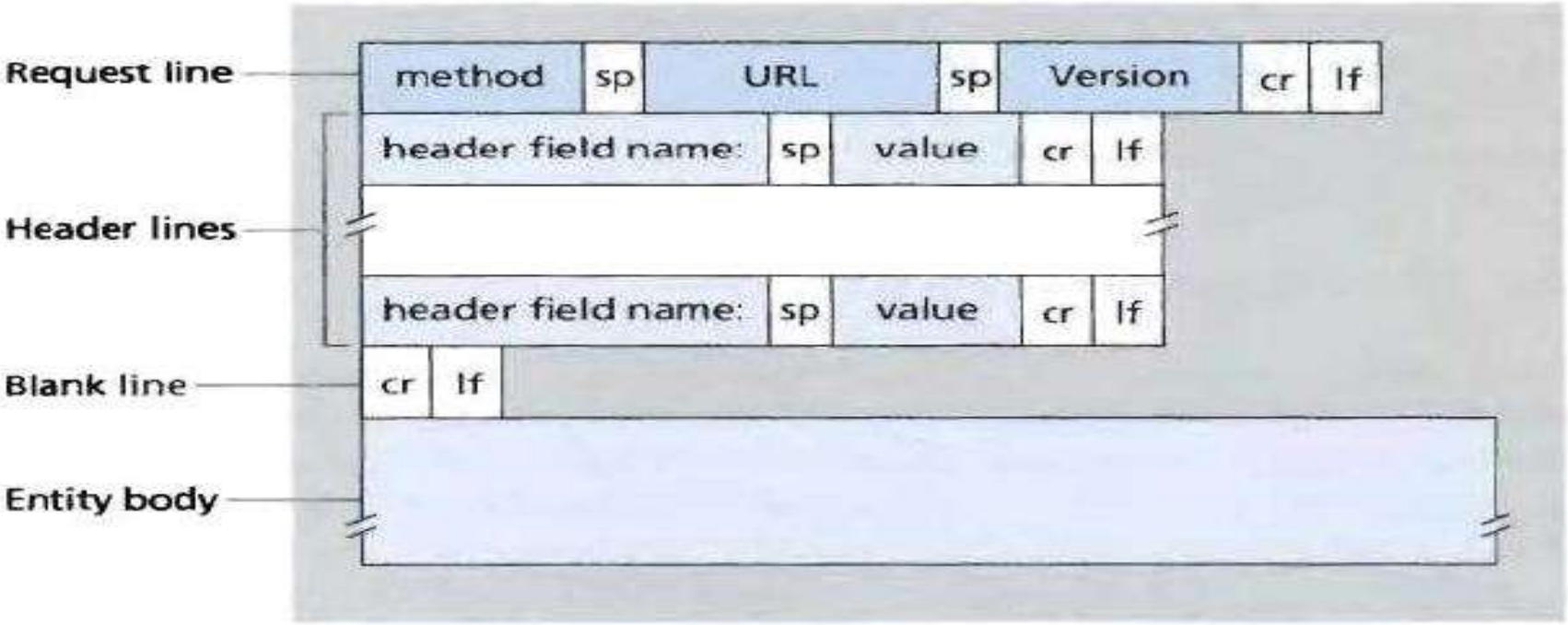


Figure 1: General format of an HTTP request message

Uploading form input

Post method:

- Web page often includes form input
- Input is uploaded to server in entity body

URL method:

- Uses GET method
- Input is uploaded in URL field of request line:

```
www.somesite.com/animalsearch?monkeys&banana
```

Method types

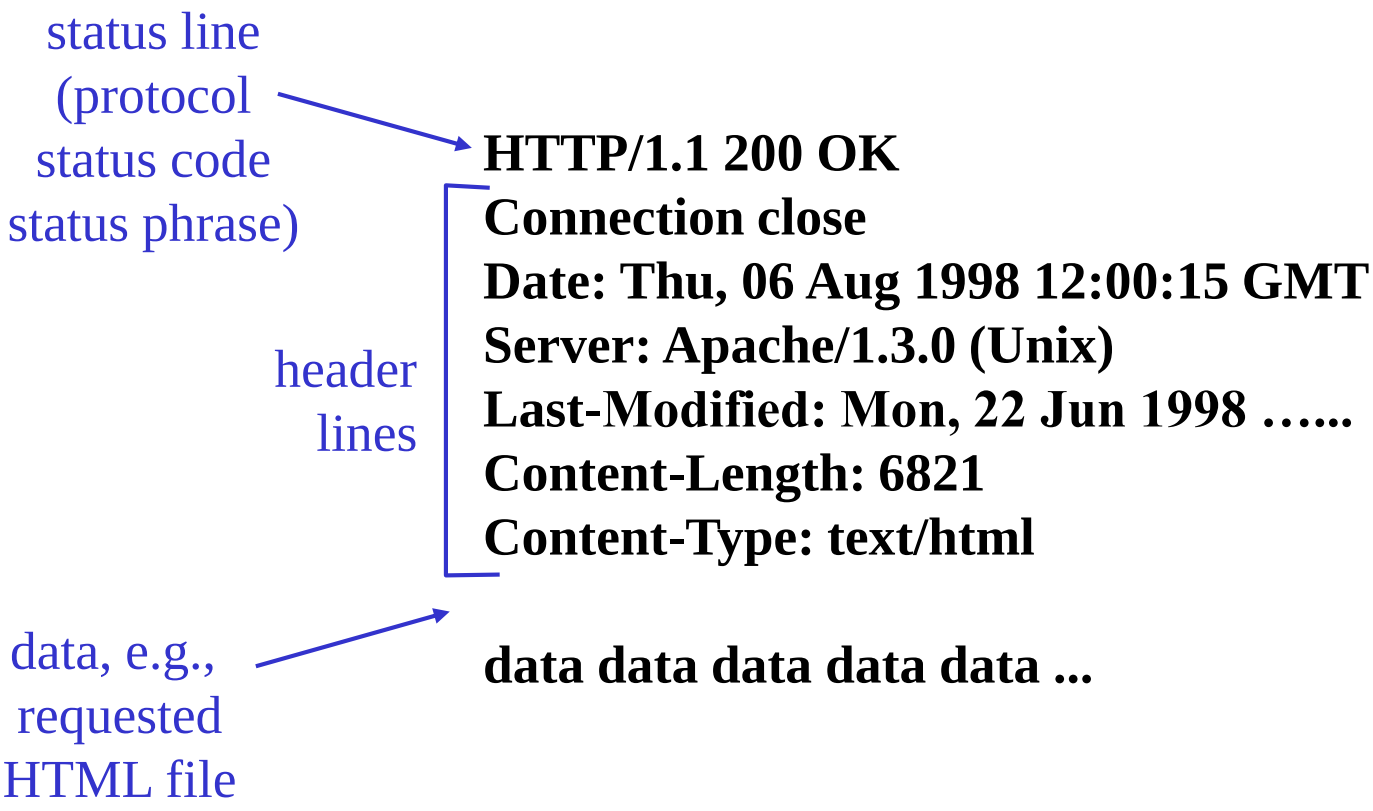
HTTP/1.0

- GET
- POST
- HEAD
  - asks server to leave requested object out of response

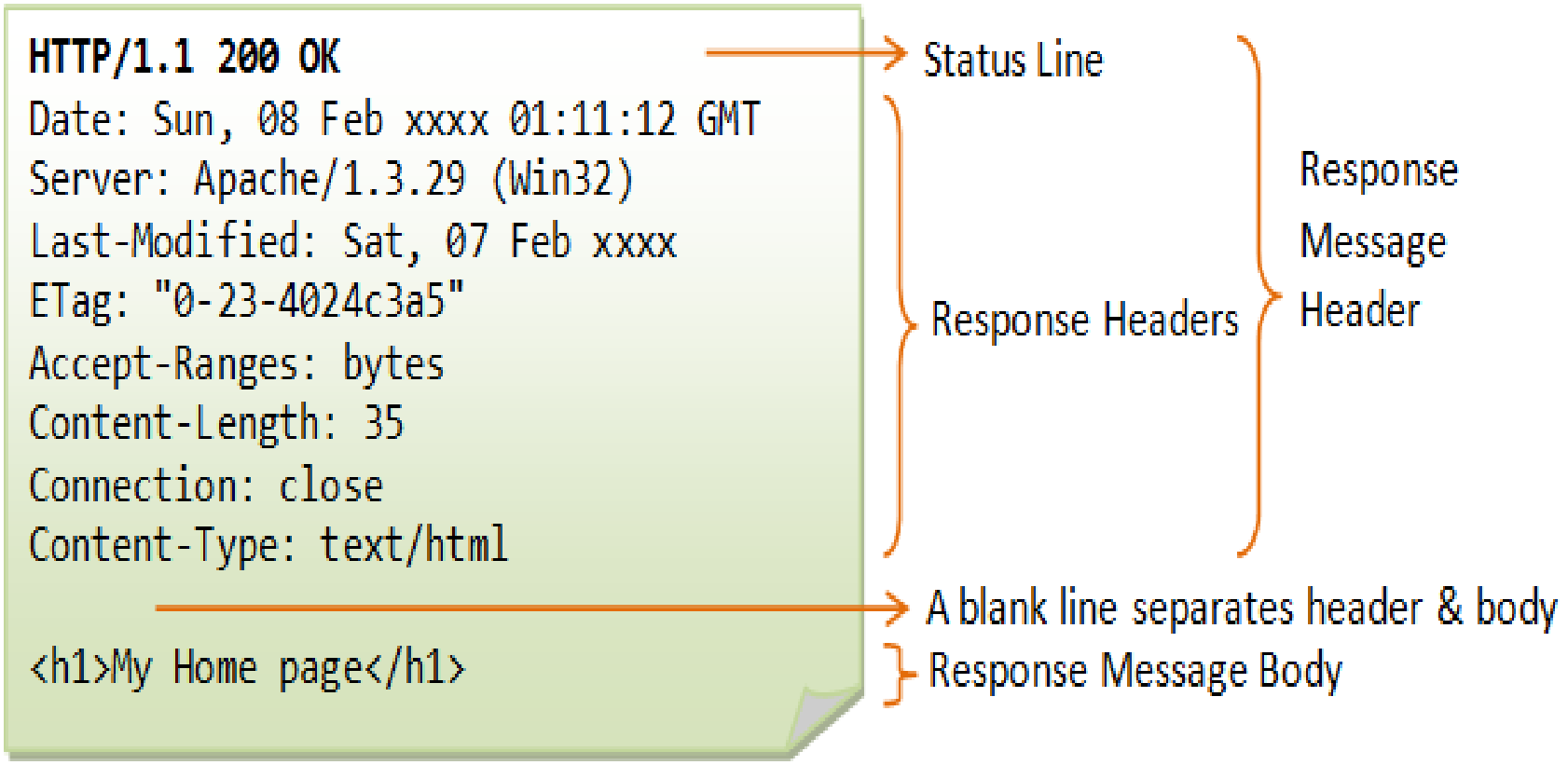
HTTP/1.1

- GET, POST, HEAD
- PUT
  - uploads file in entity body to path specified in URL field
- DELETE
  - deletes file specified in the URL field

HTTP response message



HTTP response message con..



HTTP response status codes

In first line in server->client response message.
A few sample codes:

200 OK

- request succeeded, requested object later in this message

301 Moved Permanently

- requested object moved, new location specified later in this message (Location:)

400 Bad Request

- request message not understood by server

404 Not Found

- requested document not found on this server

505 HTTP Version Not Supported

Trying out HTTP (client side) for yourself

1. Telnet to your favorite Web server:

telnet cis.poly.edu 80

Opens TCP connection to port 80 (default HTTP server port) at cis.poly.edu. Anything typed in sent to port 80 at cis.poly.edu

2. Type in a GET HTTP request:

GET /~ross/ HTTP/1.1
Host: cis.poly.edu

By typing this in (hit carriage return twice), you send this minimal (but complete) GET request to HTTP server

3. Look at response message sent by HTTP server!

User-server state: cookies

Many major Web sites use cookies

Four components:

1) cookie header line of HTTP *response* message
2) cookie header line in HTTP *request* message
3) cookie file kept on user's host, managed by user's browser
4) back-end database at Web site

User-server state: cookies

Example:

- Susan access Internet always from same PC
- She visits a specific e-commerce site for first time
- When initial HTTP requests arrives at site, site creates a unique ID and creates an entry in backend database for ID

Cookies: keeping "state"

client

server

ebay 8734

cookie file

usual http request msg

Amazon server creates ID 1678 for user

create entry

backend database

usual http response
set-cookie: 1678

ebay 8734
amazon 1678

usual http request msg
cookie: 1678

cookie-specific action

access

usual http response msg

one week later:

access

ebay 8734
amazon 1678

usual http request msg
cookie: 1678

cookie-specific action

usual http response msg

Web Cookies

What cookies can bring:

- authorization
- shopping carts
- recommendations
- user session state (Web e-mail)

How to keep "state":

- Protocol endpoints: maintain state at sender/receiver over multiple transactions
- cookies: http messages carry state

aside

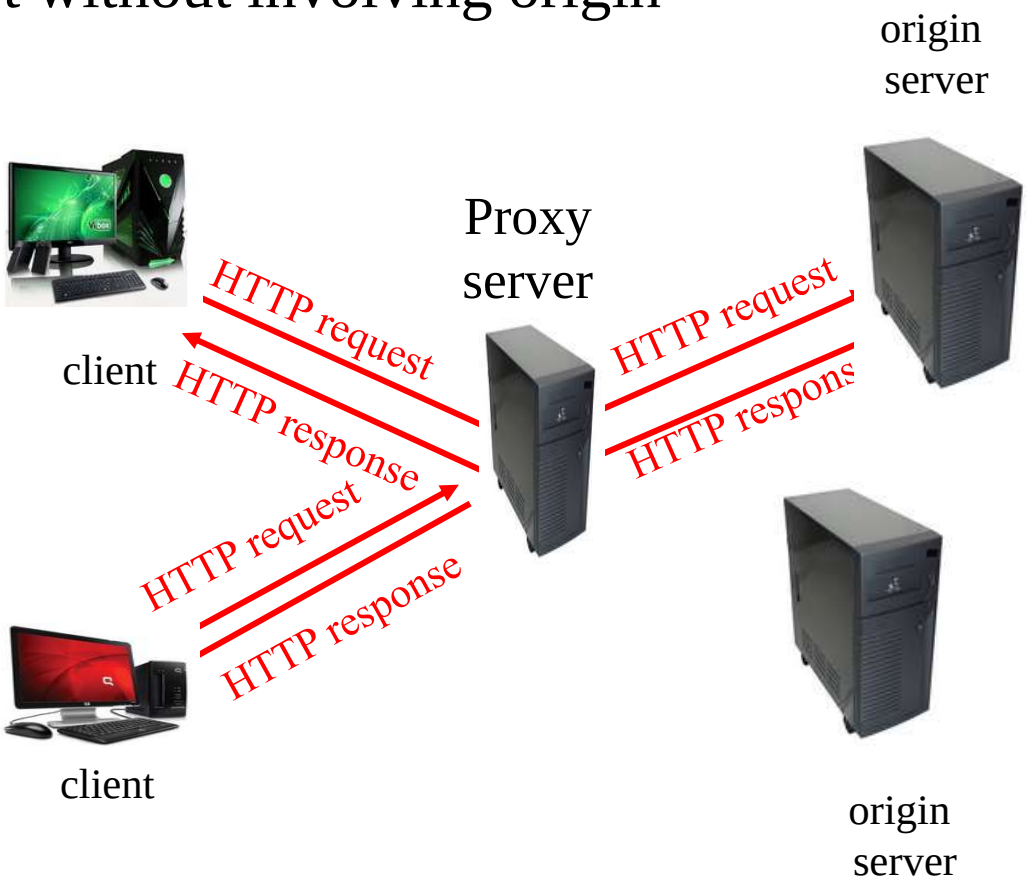
Cookies and privacy:

- cookies permit sites to learn a lot about you
- you may supply name and e-mail to sites

Web caches (proxy server)

Goal: satisfy client request without involving origin server

- user sets browser: Web accesses via cache
- browser sends all HTTP requests to cache
  - object in cache: cache returns object
  - else cache requests object from origin server, then returns object to client



More about Web caching

- Cache acts as both client and server
- Typically cache is installed by ISP (university, company, residential ISP)

Why Web caching?

- Reduce response time for client request.
- Reduce traffic on an institution's access link.
- Internet dense with caches: enables "poor" content providers to effectively deliver content (but so does P2P file sharing)