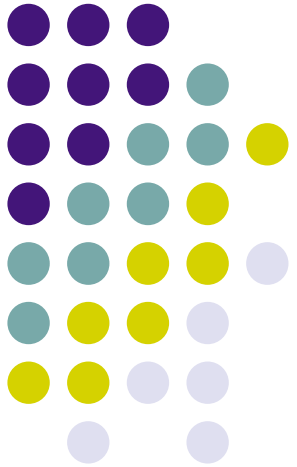
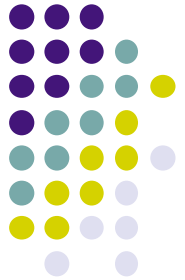
Lecture 9: Application layer

Reading Chapter 7

Computer networks, Tanenbaum

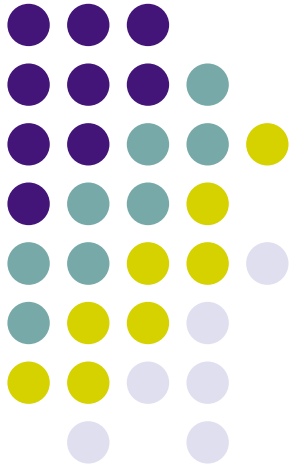


Contents

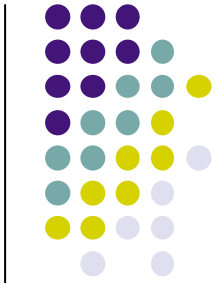


- Application layer
  - Fundamental concepts
  - Case study: HTTP, Mail, FTP…

Fundamental concepts



Application layer in OSI model

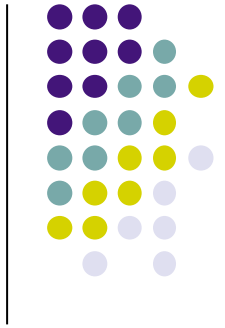


Application (HTTP, Mail, …)
Transport (UDP, TCP …)
Network (IP, ICMP…)
Datalink (Ethernet, ADSL…)
Physical (bits…)

Protocols communication between parties of the application

Transmission data between application

Application and service?



MUSIC ONLINE

VoIP

GAME
ON LINE

CHAT

VoD

e-Office

SMS

e-BANK

MAIL

E-learning

WEB

YOUTUBE

VIDEO
CONFERENCE

FTP

EBAY

GOOGLE

SKYPE

Social
networks

SSH

NEWS

BITTORENT

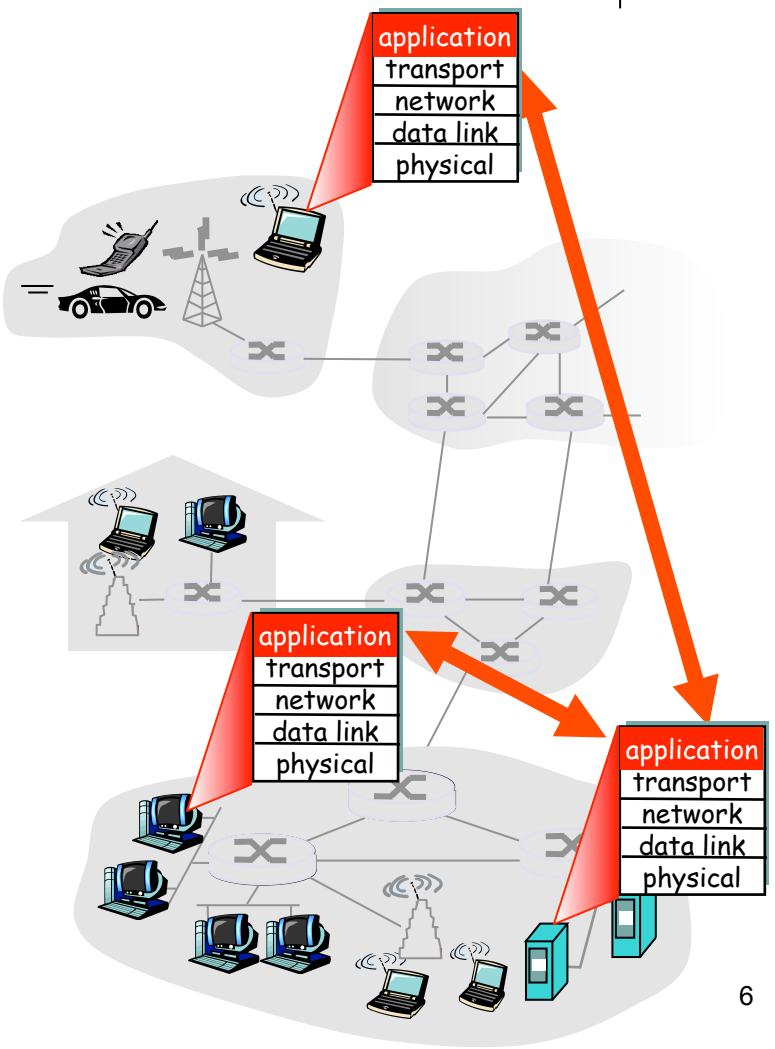
E-COMMERCE

GRID

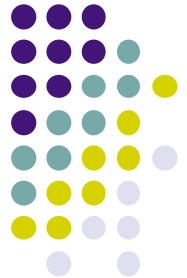
e-Goverment

Application and application protocol

- Application protocol
  - Define communication rule
  - Use service of transport layer (TCP/UDP…)
- Application:
  - Is a process on the internet. They communicate to each other by exchanging messages.
  - Runs on end systems
  - Use application protocol for providing service
- Example of application/ protocol:
  - Web (HTTP)
  - Mail (SMTP/POP/IMAP) …



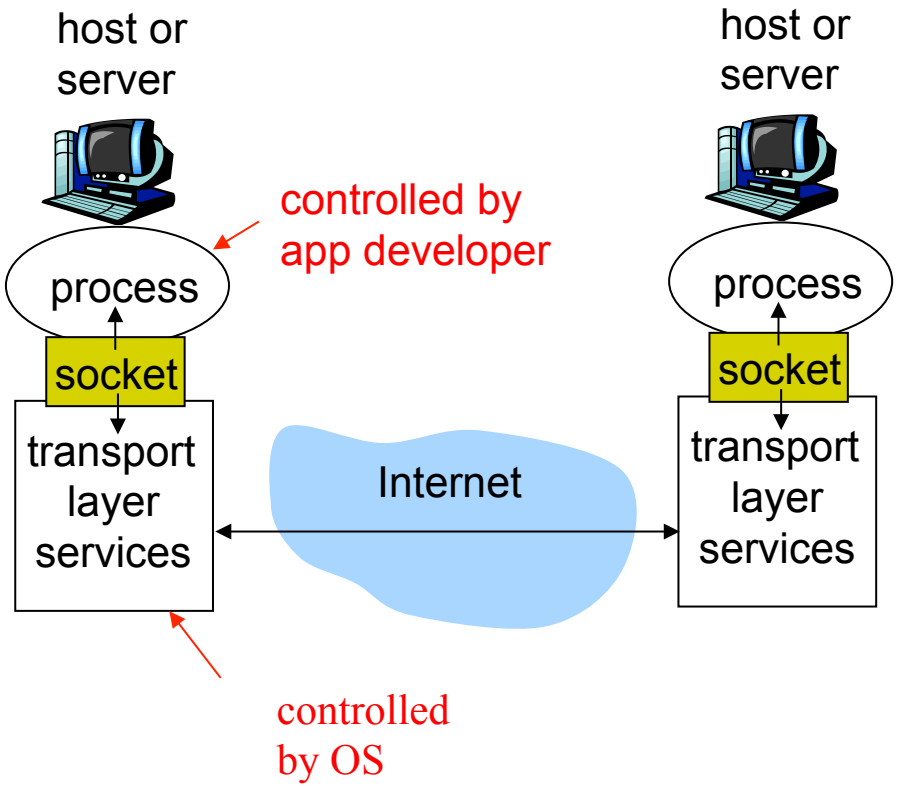
Components of an application



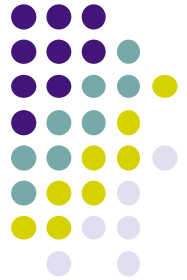
- Application software is compose of
  - User interface:
    - Interfacing with users,
    - e.g. Web browser (Firefox, IE), mail reader(Thunderbird, Outlook,..)
    - Implement one part of application protocol
  - Server program:
    - Cung cấp dịch vụ cho người sử dụng
- Application process: the application software running on an OS

Communication between process on the Internet

- Socket is an interface between an application process and transport layer
- Socket is defined by
  - Số hiệu cổng
  - Địa chỉ IP
  - Kiểu giao thức giao vận (TCP hay UDP)
- Socket API (Application Programming Interface): Allow application to choose parameters for transport service
  - Choose transport protocol
  - Type of IO communication …

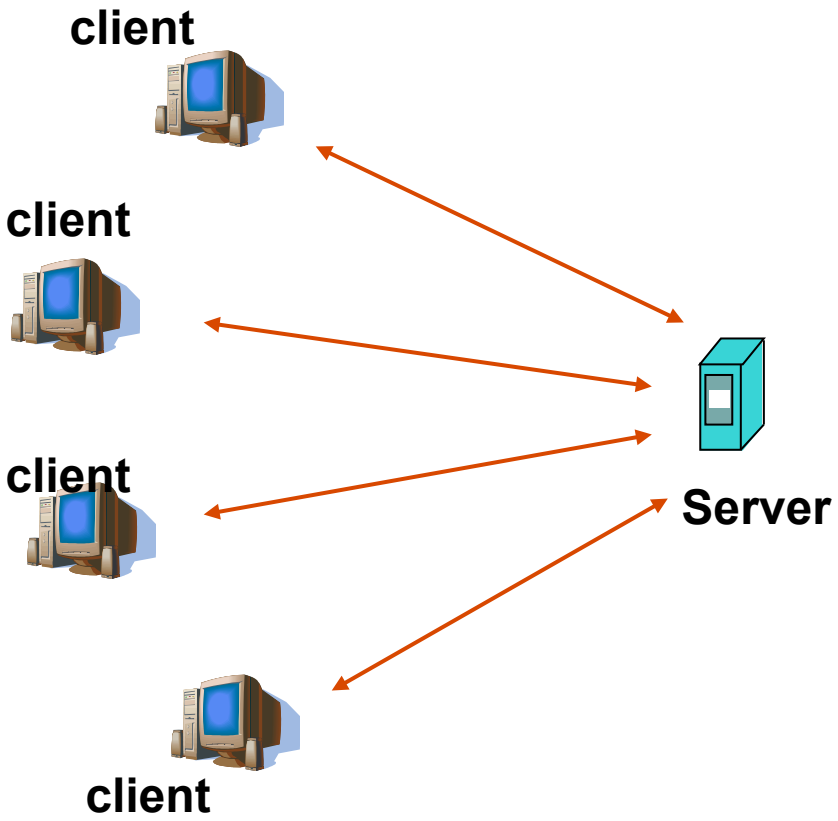


Application architecture



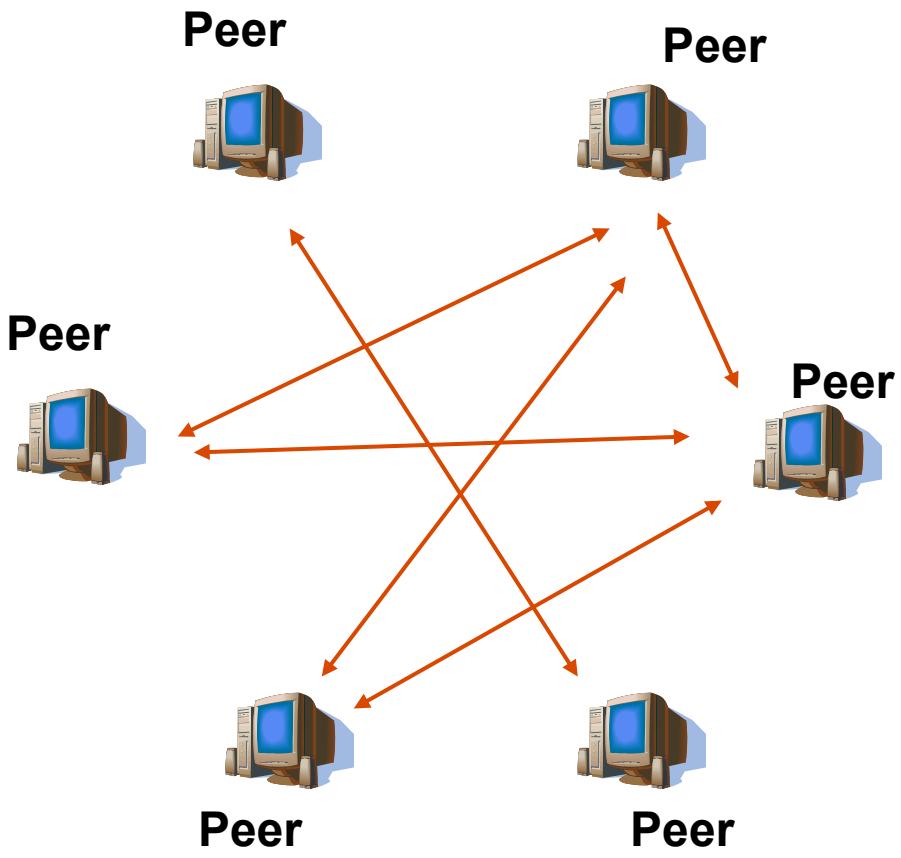
- Client-server
- P2P
- Hybrid

Client-server



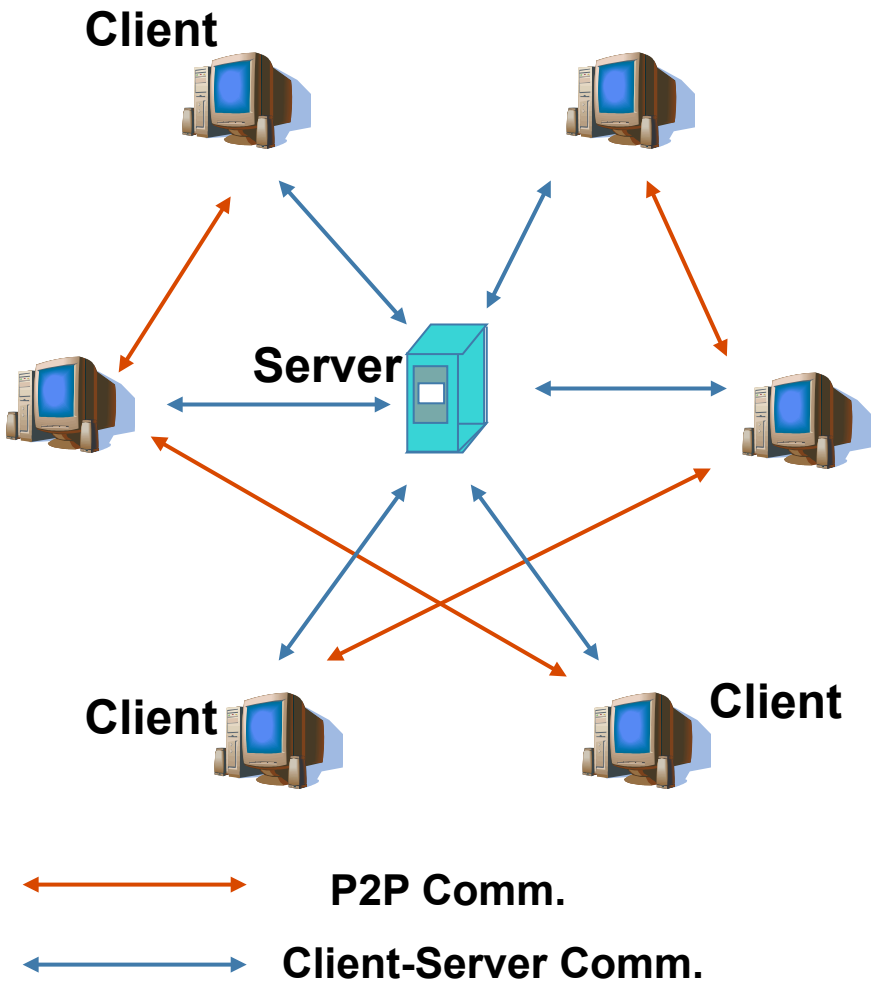
- Two kind of components: client and server
- Client
  - Client sends requests for service to server
  - Clients do not contact directly to each other
- Server
  - Always online waiting for service requests from clients
  - There may be backup servers for assuring high availability in failures
- e.g. Web, Mail, …

Pure Peer-to-peer architecture



- No center server, only peers as components
- Peers have equal role in the system
- Any two peers can communicate directly to each other but only when both are online.
- Peer does not need to be online all the time
- E.g. Gnutella, Bittorent

Hybrid architecture

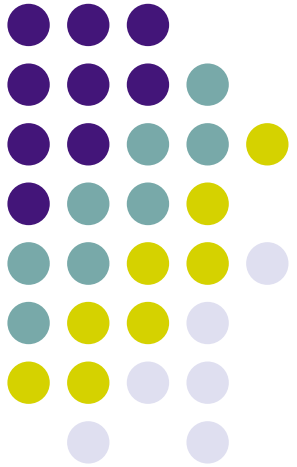


- A center server for user management, indexing for search purpose.
- Clients communicate directly to each other after authentication process with server.
- E.g. Skype (before 2016)
  - Skype server manage user lists, authentification
  - After authentification users communicate directly to each other

Case study 1: HTTP and WWW

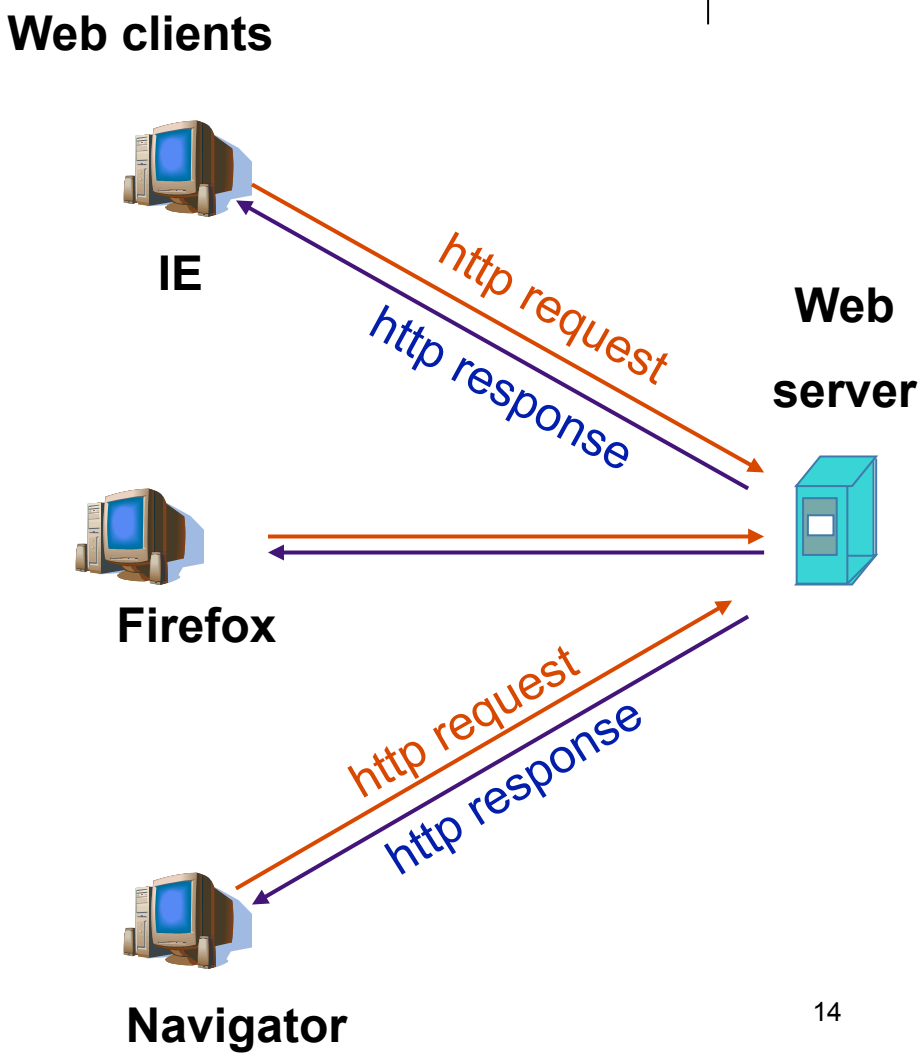
Reading 7.3

Computer Networks, Tanenbaum

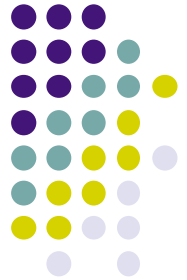


HTTP and Web

- WWW: World Wide Web
  - Application for exchanging the HTML documents (HyperText Markup Language) over Internet
  - WWW use HTTP protocol
- HTTP: HyperText Transfer Protocol
  - Client/Server model
  - Client (Web browser)requests for webpages and displays them on its interface
  - Server: Receive request from client and return results under the form of webpage.



How HTTP works?



- Server open a TCP socket by default at port 80 waiting for
- Client initiates a TCP connection to server
- Server accept the connection request
- Exchange HTTP message
  - HTTP Request
  - HTTP Response
- Close connection TCP

Format of HTTP request

- ASCII encoding (readable using text editor)

request line (GET, POST, HEAD commands)

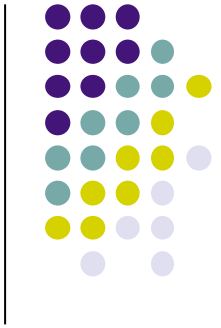
header lines

```
GET /dccn/index.html HTTP/1.1
Host: www.it-hut.edu.vn
User-agent: Mozilla/4.0
Connection: close
Accept-language:en-us
```

CR, LF → (extra carriage return, line feed)

indicates end of message

Format of HTTP response



status line (protocol status code status phrase)

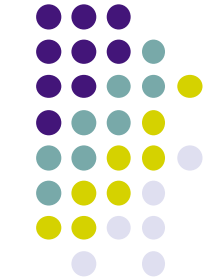
header lines

data, e.g., requested HTML file

```
HTTP/1.1 200 OK
Connection close
Date: Tue, 16 Mar 2008 12:00:15 GMT
Server: Apache/1.3.0 (Unix)
Last-Modified: Mon, 15 Mar 2008 …...
Content-Length: 8990
Content-Type: text/html

data data data data data ...
```

Types of HTTP connections



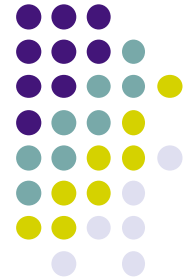
HTTP non-persistent

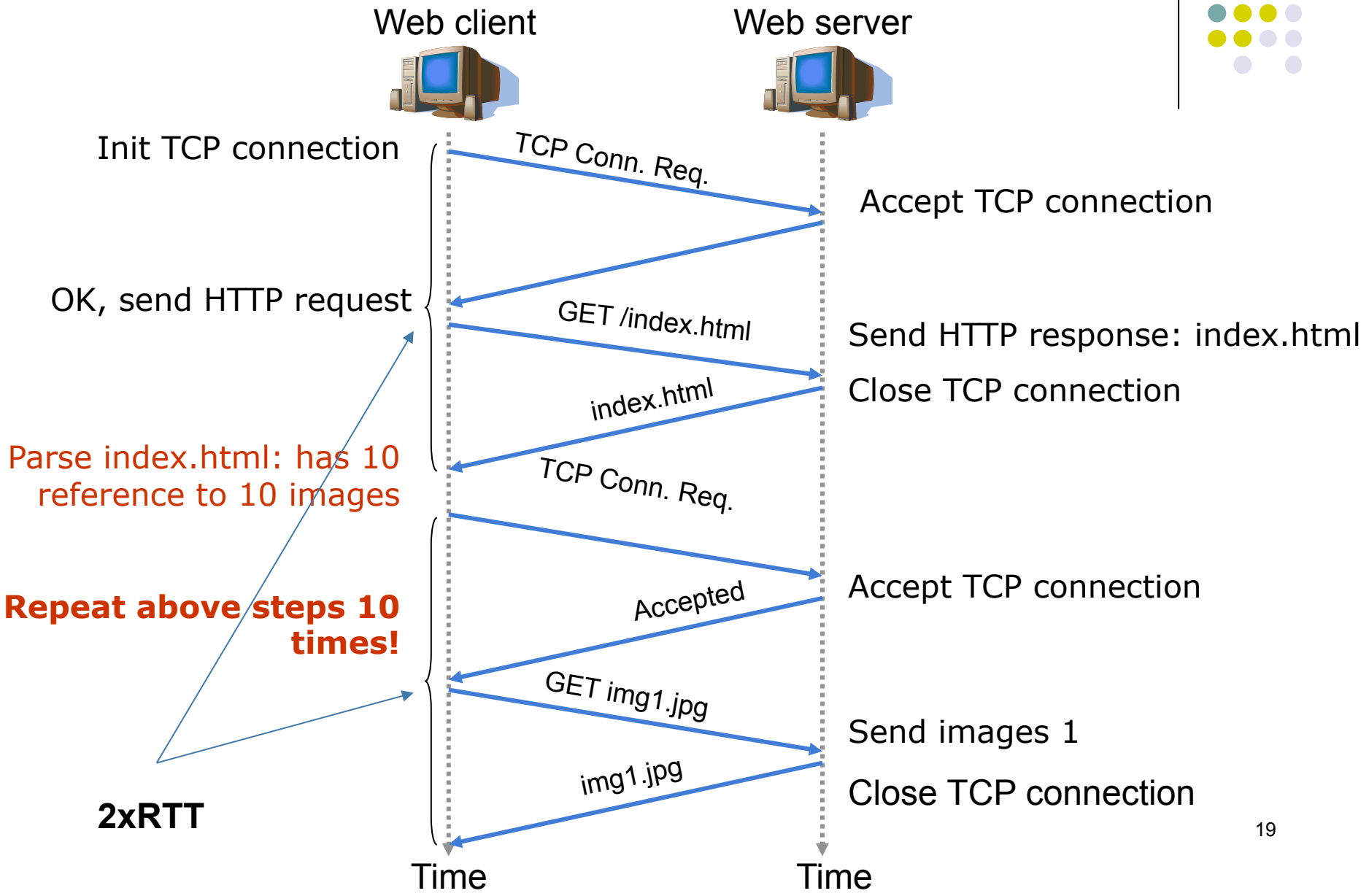
- Only one web object (text or image) is transferred over a connection TCP
- Option by default in HTTP/1.0
- HTTP 1.0: RFC 1945

HTTP persistent

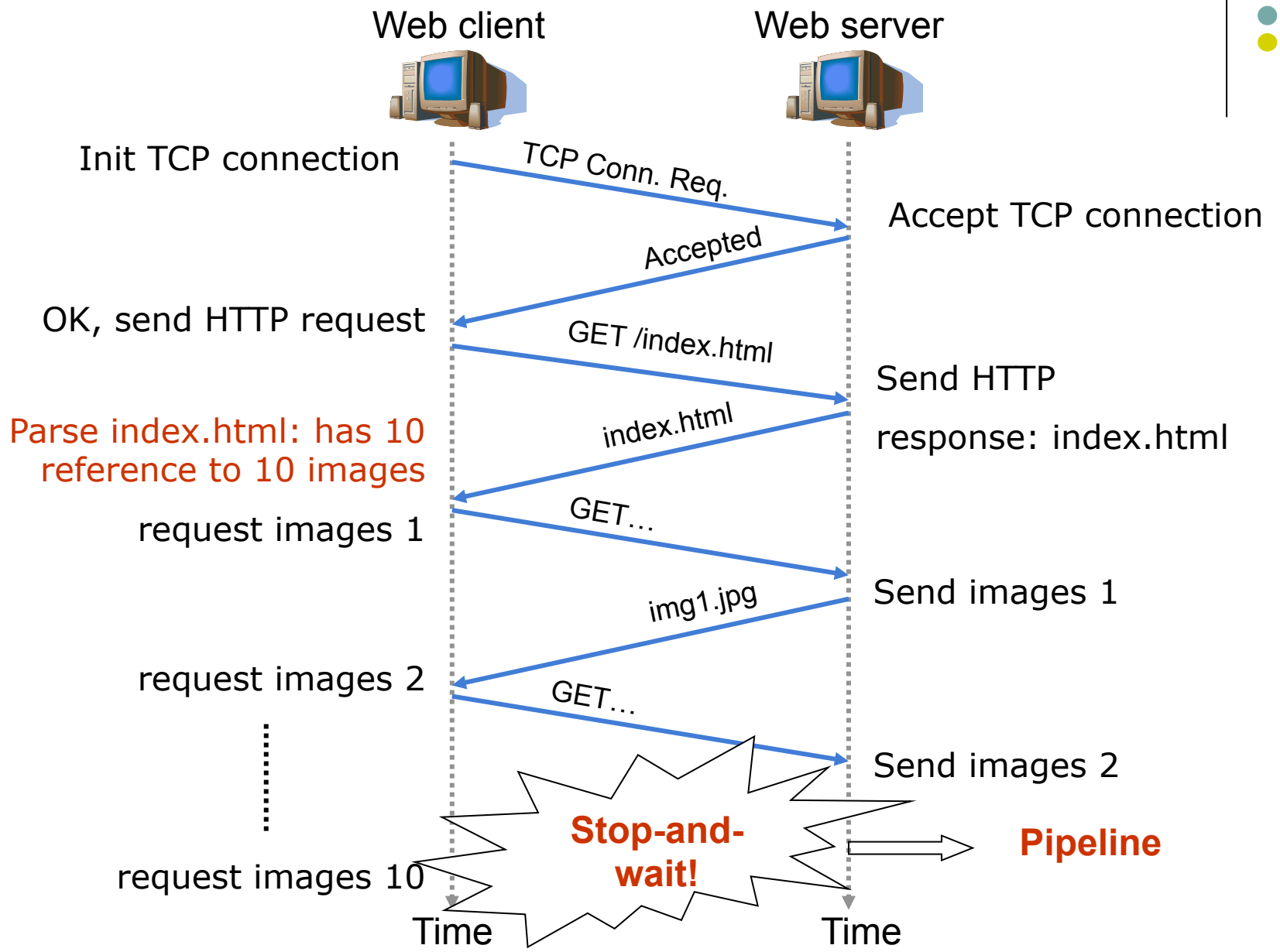
- Many web objects can be sent over a connection TCP.
- Option by default in HTTP/1.1
- HTTP 1.1: RFC 2068

Operation of HTTP/1.0

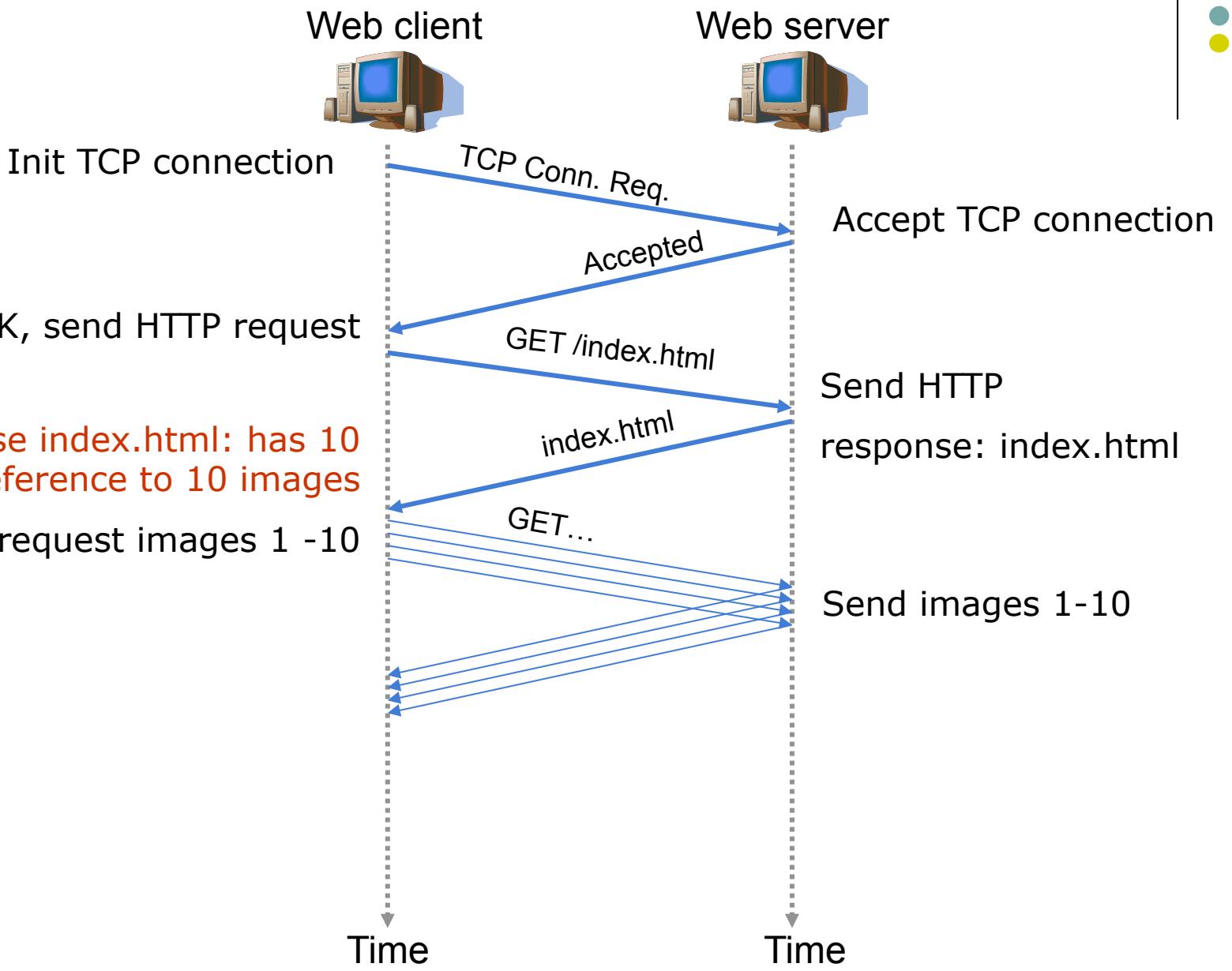




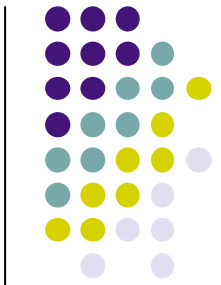
Operation of HTTP/1.1



HTTP/1.1 with pipeline



Methods in HTTP request message



HTTP/1.0

- GET: get an webpage
- POST: submitting a form
- HEAD: ask for the header of an webpage

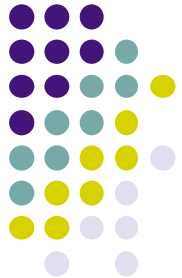
HTTP/1.1

- GET, POST, HEAD
- PUT
  - Upload an webpage to the server under address given in URI, file content is in the body of the message
- DELETE
  - Delete a file given in the URI

Attention: Even with GET, user can sends parameters to servers in URL. Ex:

http://www.google.com/search?q=computer+network&flags=68&num=10

Status code in Response message



Status code is in the first line of the Response message

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