HTTP AND THE WEB

George Porter
Module 1
Fall 2020

UC San Diego

ATTRIBUTION

- These slides are released under an Attribution-NonCommercial-ShareAlike 3.0 Unported (CC BY-NC-SA 3.0) Creative Commons license
- These slides incorporate material from:
    - Computer Networks: A Systems Approach, 5e, by Peterson and Davie

HTTP AS AN EMERGING TRANSPORT LAYER

- HTTP: HyperText Transfer Protocol

  - Tim Berners-Lee at CERN in 1989

  - Used for web browsing

- In addition to web browsing:

  - Video streaming via DASH on YouTube/Netflix, etc

  - REST (Representational state transfer)

 - Chat apps like Slack

  - Many others
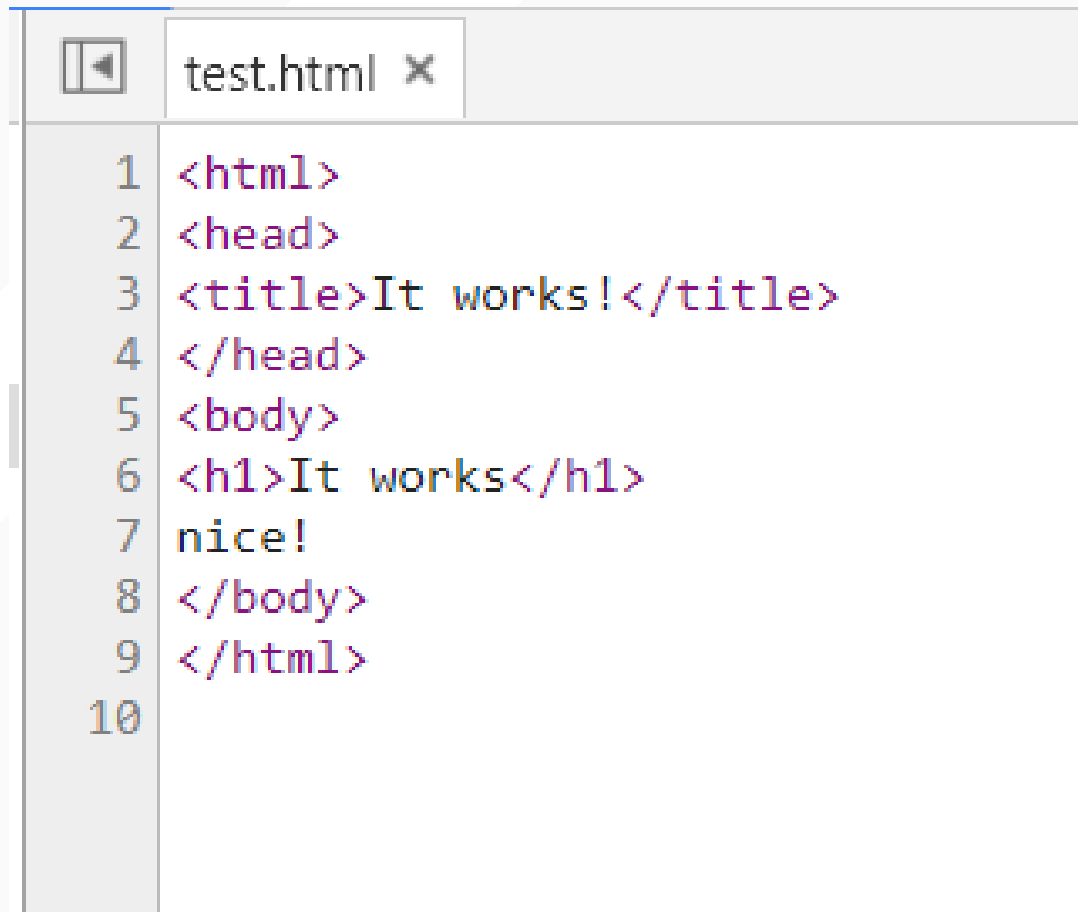
WEB/HTTP OVERVIEW

- Documents link to other documents

  - Specified in HTML files

- HTTP is the protocol for retrieving HTML files from servers

  - and images, sounds, video, …

- Implemented in servers

  - Apache, nginx, MSFT IIS

- and clients

  - Chrome

  - MSFT Edge

  - Apple Safari…

Request HTML page

<HTML>

Parses HTML

Request image

<Image>

Request video

<Video>
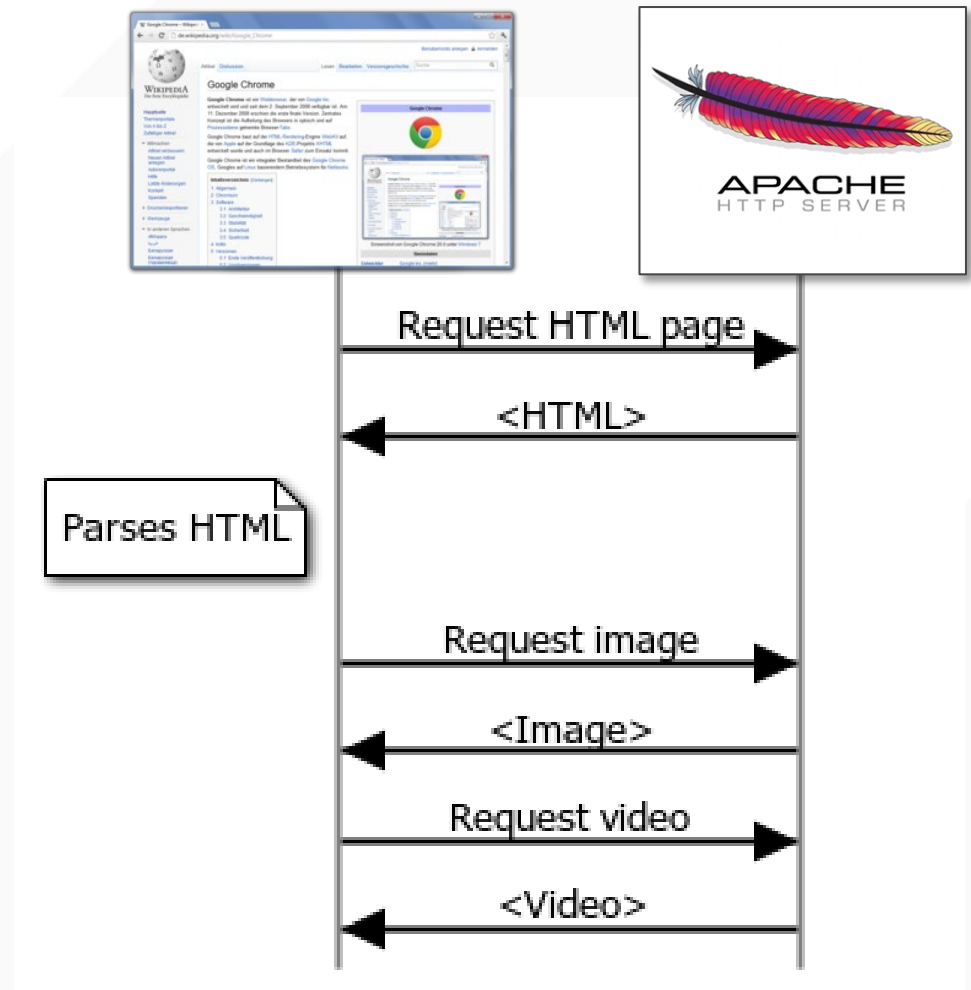
AGGREGATING CONTENT FROM WEB SERVERS
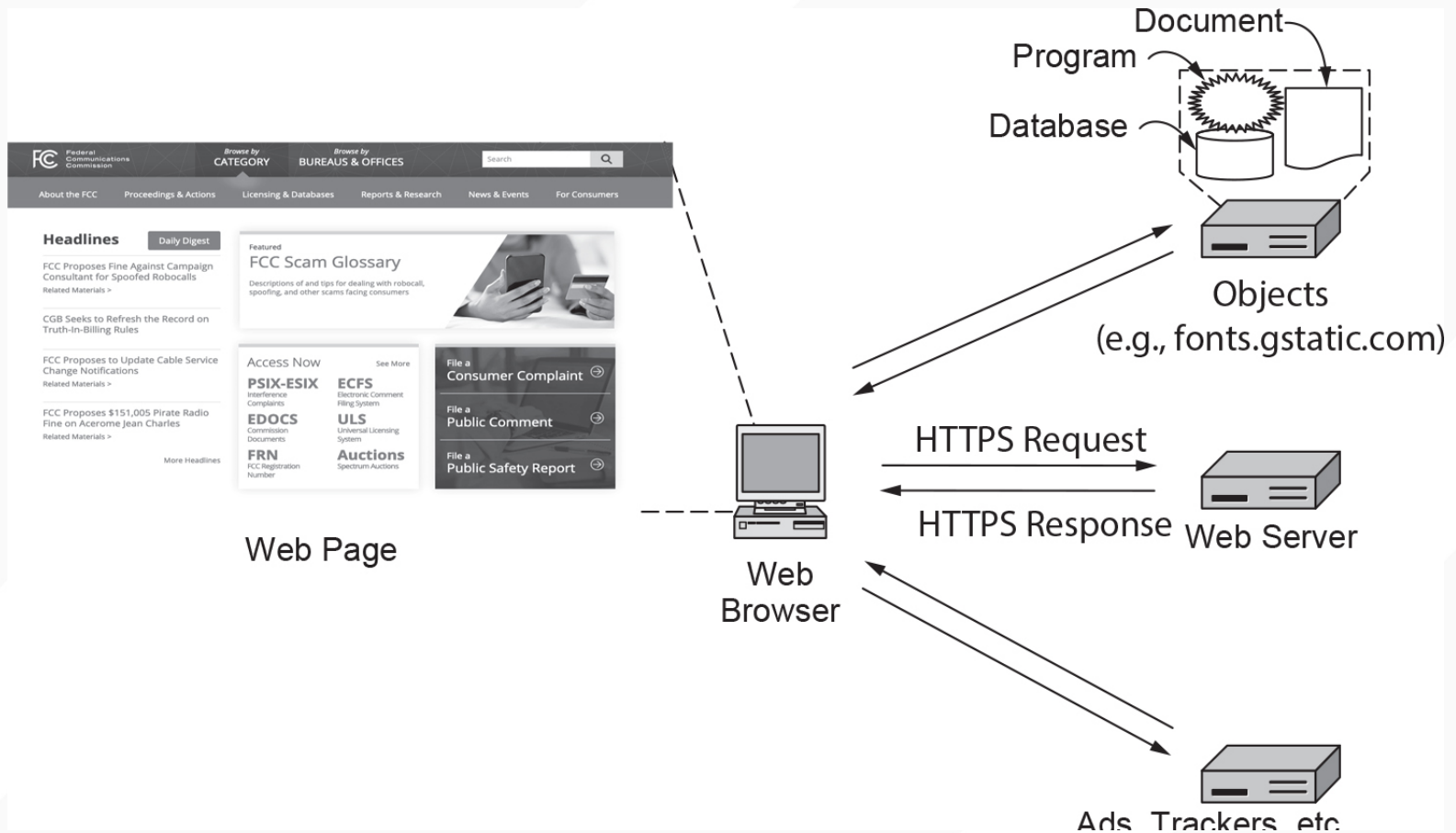
SAMPLE HTML FILE

```
test.html ✕

1  <html>
2  <head>
3  <title>It works!</title>
4  </head>
5  <body>
6  <h1>It works</h1>
7  nice!
8  </body>
9  </html>
10
```

A WEB REQUEST EXAMINED

- Steps that occur when a link is selected:

  - Browser determines the URL

  - Browser asks DNS for the IP address of the server

  - DNS replies

  - Browser opens a TCP connection

  - Sends HTTP request for the page

  - Server sends the page as HTTP response

  - Browser fetches other URLs as needed

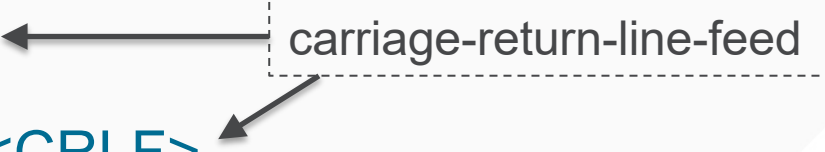 - Browser displays the page

  - The TCP connections are released

EXAMPLE OF REQUESTS MAKING UP FCC.GOV

HTTP OVERVIEW

- HTTP is a text oriented protocol.

- HTTP is a request/response protocol

- Requests and responses both look like:

  START_LINE <CRLF> ← carriage-return-line-feed

  MESSAGE_HEADER <CRLF>

  <CRLF>

  MESSAGE_BODY <CRLF>

- The first line (START LINE) indicates whether this is a request message or a response message.

HTTP REQUESTS

- Request Messages define

  - The operation (called *method*) to be performed

  - The web page the operation should be performed on

  - The version of HTTP being used.

- Examples:

  - GET /index.html HTTP/1.0

  - GET /images/catimg23.jpg HTTP/1.1

  - GET /contracts/contract3.txt HTTP/1.1

HTTP METHODS

Method	Description
GET	Read a Web page
HEAD	Read a Web page's header
POST	Append to a Web page
PUT	Store a Web page
DELETE	Remove the Web page
TRACE	Echo the incoming request
CONNECT	Connect through a proxy
OPTIONS	Query options for a page

OPTIONAL HTTP REQUEST HEADERS

- After the start line are *request headers*:

  - Text-based, key and value separated by a colon

- Example 1:

  GET /index.html HTTP/1.0

  User-Agent: Firefox 23.3.1

- Example 2:

  GET /images/cat2.jpg HTTP/1.1

  Host: www.cs.ucsd.edu

  User-Agent: Chrome 12.1

HTTP HEADERS

Header	Type	Contents
User-Agent	Request	Information about the browser and its platform
Accept	Request	The type of pages the client can handle
Accept-Charset	Request	The character sets that are acceptable to the client
Accept-Encoding	Request	The page encodings the client can handle
Accept-Language	Request	The natural languages the client can handle
If-Modified-Since	Request	Time and date to check freshness
If-None-Match	Request	Previously sent tags to check freshness
Host	Request	The server's DNS name
Authorization	Request	A list of the client's credentials
Referrer	Request	The previous URL from which the request came
Cookie	Request	Previously set cookie sent back to the server
Set-Cookie	Response	Cookie for the client to store
Server	Response	Information about the server
Content-Encoding	Response	How the content is encoded (e.g., *gzip*)
Content-Language	Response	The natural language used in the page
Content-Length	Response	The page's length in bytes
Content-Type	Response	The page's MIME type
Content-Range	Response	Identifies a portion of the page's content
Last-Modified	Response	Time and date the page was last changed
Expires	Response	Time and date when the page stops being valid
Location	Response	Tells the client where to send its request
Accept-Ranges	Response	Indicates the server will accept byte range requests
Date	Both	Date and time the message was sent
Range	Both	Identifies a portion of a page
Cache-Control	Both	Directives for how to treat caches
ETag	Both	Tag for the contents of the page
Upgrade	Both	The protocol the sender wants to switch to

HTTP RESPONSES

- Also begins with a single START LINE.

  - The version of HTTP being used, A three-digit status code, text description of the code

- Example:

 HTTP/1.1 200 OK

  Content-Type: text/html

  Content-Length: 291

Code	Meaning	Examples
1xx	Information	100 = server agrees to handle client's request
2xx	Success	200 = request succeeded; 204 = no content present
3xx	Redirection	301 = page moved; 304 = cached page still valid
4xx	Client error	403 = forbidden page; 404 = page not found
5xx	Server error	500 = internal server error; 503 = try again later

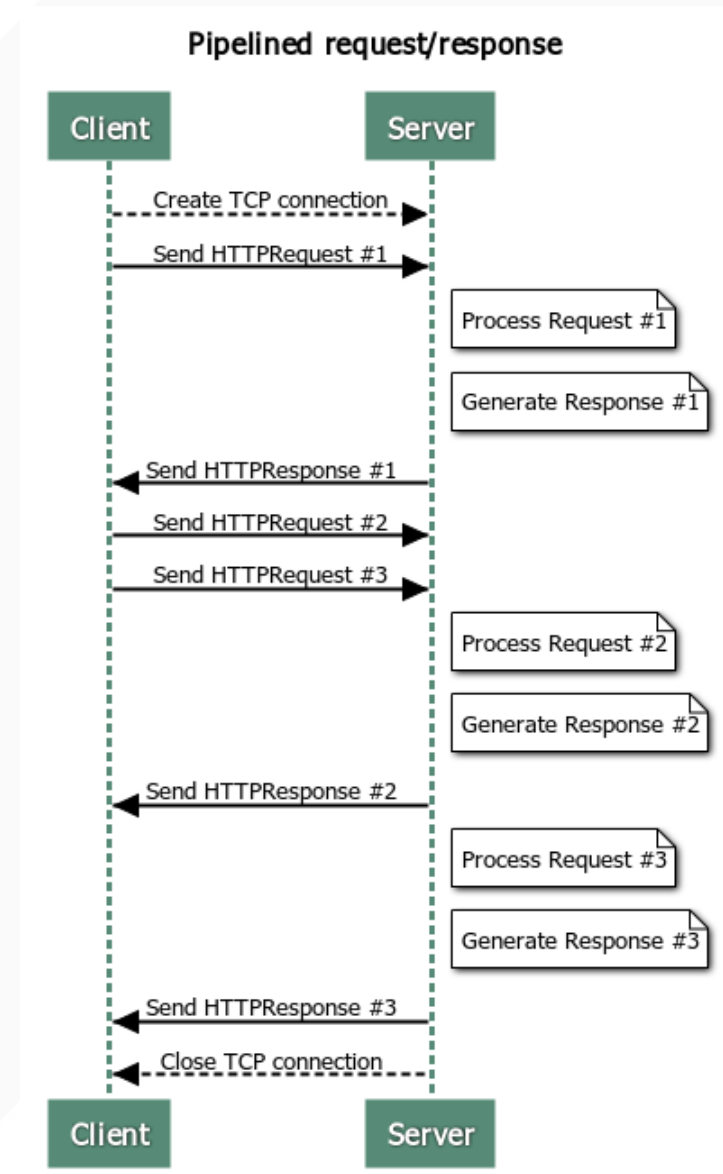# DESIGNING YOUR SERVER

- Steps the server performs in its main loop:

  - Accept a TCP connection from a client (a browser)

  - Get the path to the page, which is the name of the file requested

  - Get the file (from disk)

  - Send an HTTP header then contents of the file to the client

  - Release the TCP connection

- Modern Web servers have more features

- For dynamic content

  - Third step may be replaced by the execution of a program that generates and returns the contents
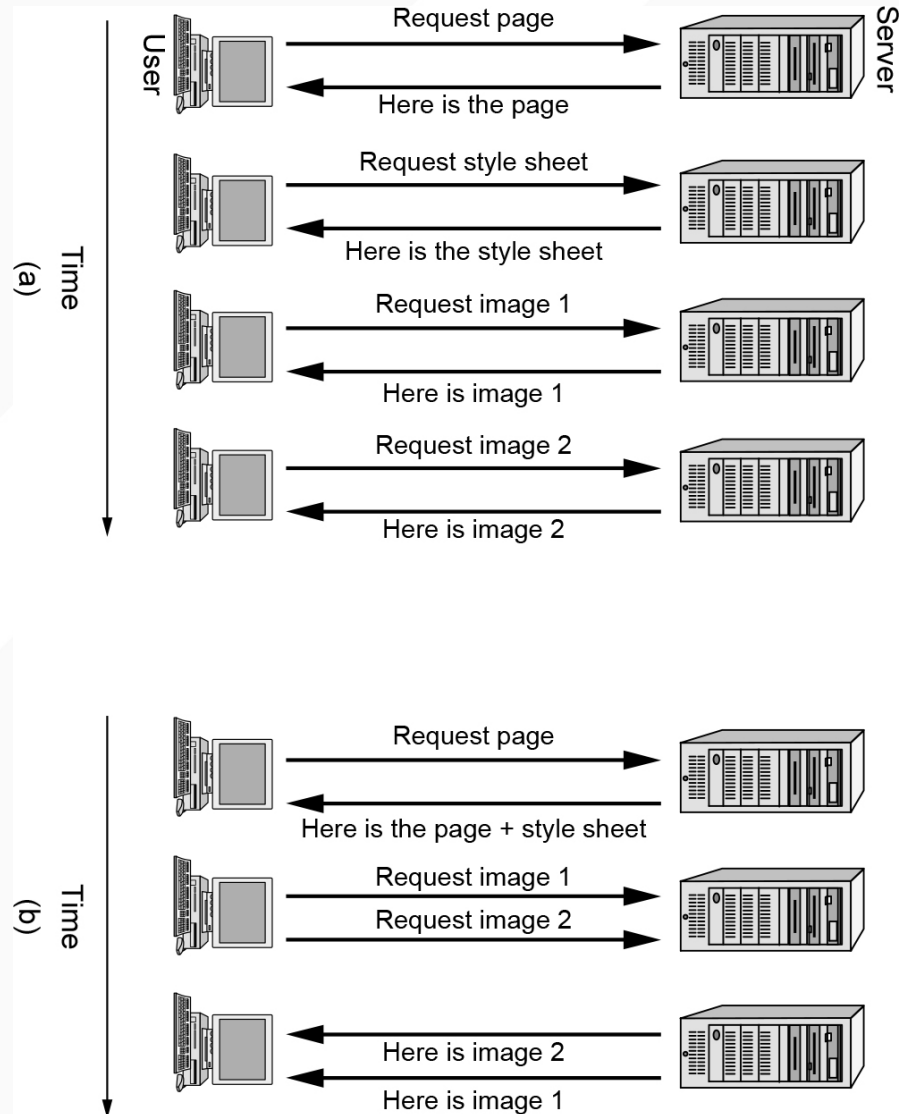
TYPICAL SERVER DESIGN
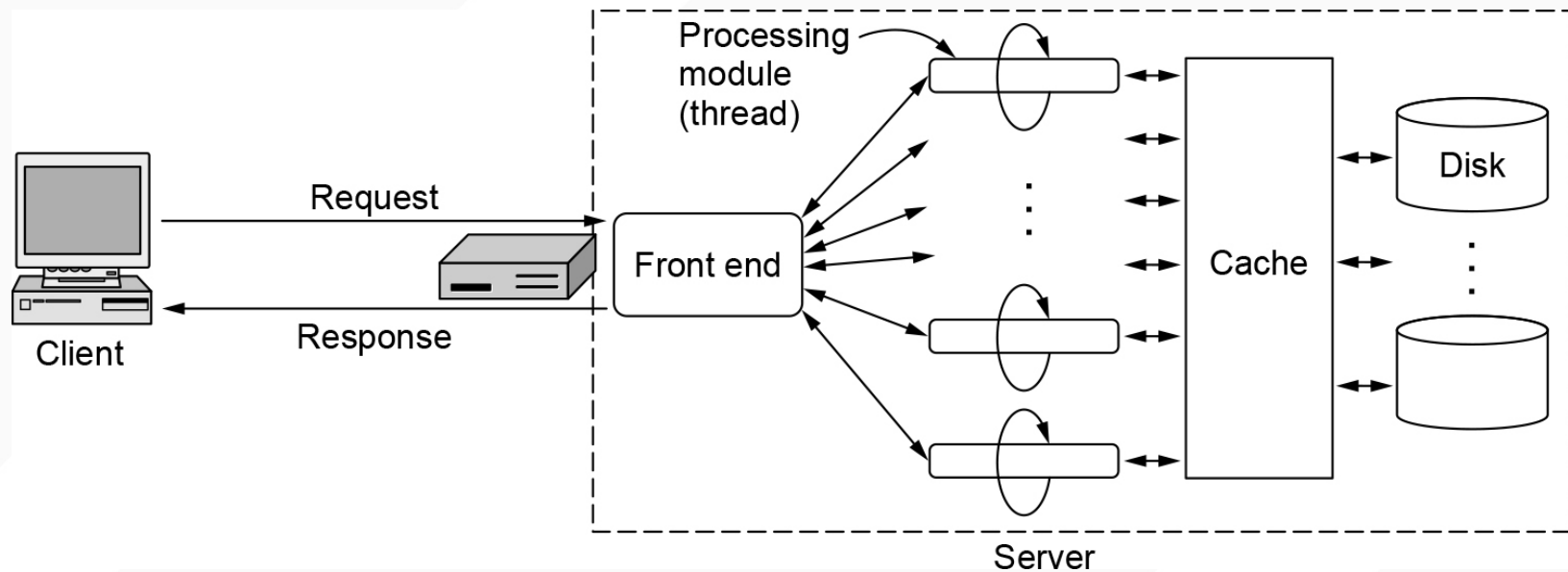
HTTP PIPELINING (VERSION HTTP/1.1)

- HTTP/1.0 opened a new connection for every data item it retrieved

- Overhead in establishing a new connection to the same server over and over again

- HTTP/1.1 Persistent Connections

  - Reuse connection over many requests/responses

  - But more complex in terms of framing/parsing

    - How to know when one request ends and the next begins?

    - This is part of the 1.1 spec

Pipelined request/response

EXAMPLE OF PIPELINING

(a) Multiple connections and sequential requests.
(b) A persistent connection and pipelined requests.