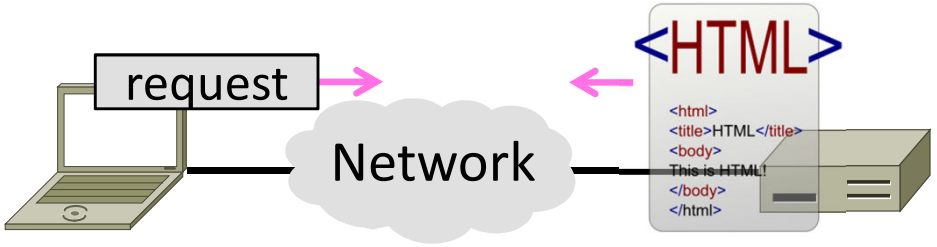
# Introduction to Computer Networks

## HTTP, the HyperText Transfer Protocol (§7.3.1-7.3.4)

David Wetherall (djw@uw.edu)
Professor of Computer Science & Engineering
UNIVERSITY of WASHINGTON

# Topic

- HTTP, (HyperText Transfer Protocol)
  - Basis for fetching Web pages

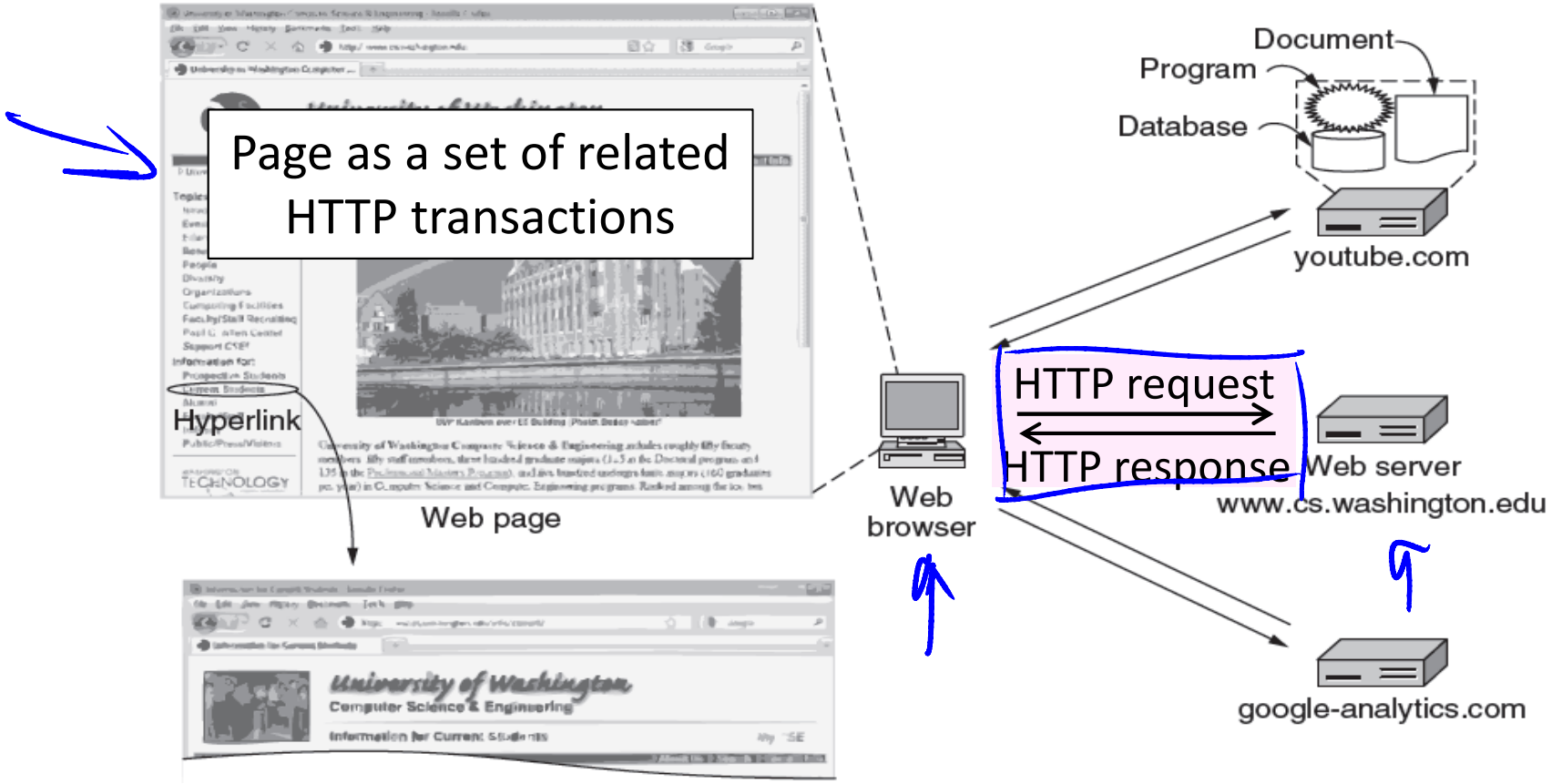


# Sir Tim Berners-Lee (1955–)

- Inventor of the Web
  - Dominant Internet app since mid 90s
  - He now directs the W3C
- Developed Web at CERN in ‘89
  - Browser, server and first HTTP
  - Popularized via Mosaic (‘93), Netscape
  - First WWW conference in ’94 …

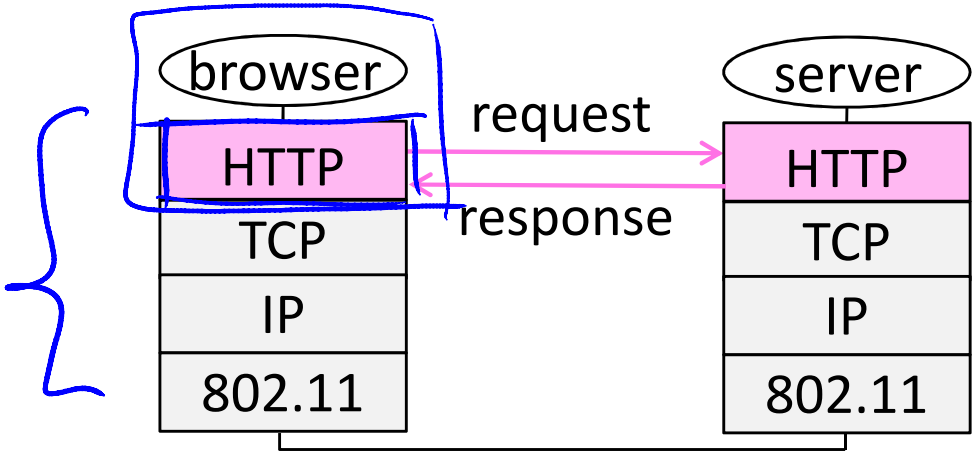
Source: By Paul Clarke, CC-BY-2.0, via Wikimedia Commons

# Web Context


Page as a set of related HTTP transactions
Hyperlink
Web page
Document
Program
Database
youtube.com
HTTP request
HTTP response
Web browser
Web server
www.cs.washington.edu
google-analytics.com
University of Washington
Computer Science & Engineering

# HTTP Context

- HTTP is a request/response protocol for fetching Web resources
  - Runs on TCP, typically port 80
  - Part of browser/server app



# Fetching a Web page with HTTP

- Start with the page URL:

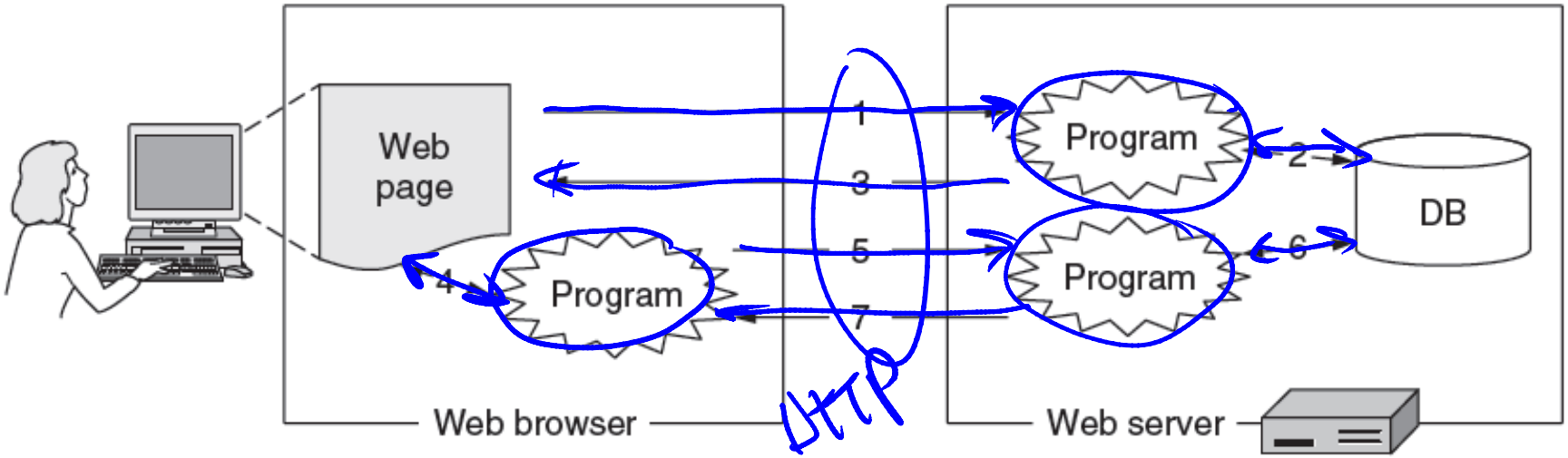
  http://en.wikipedia.org/wiki/Vegemite

  Protocol  Server  Page on server

- Steps:
  - Resolve the server to IP address (DNS)
  - Set up TCP connection to the server
  - Send HTTP request for the page
  - (Await HTTP response for the page)
  - ** Execute / fetch embedded resources / render
  - Clean up any idle TCP connections

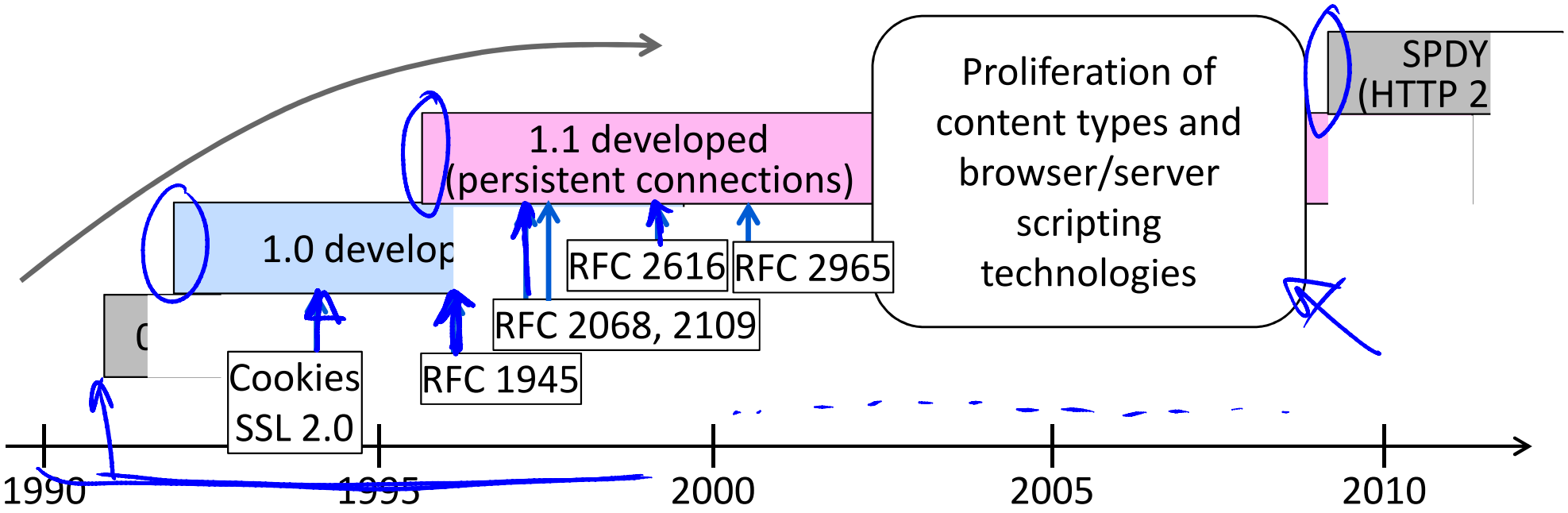
# Static vs Dynamic Web pages

- Static web page is a file contents, e.g., image
- Dynamic web page is the result of program execution
  - Javascript on client, PHP on server, or both



# Evolution of HTTP

- Consider security (SSL/TLS for HTTPS) later



# HTTP Protocol

- Originally a simple protocol, with many options added over time
  - Text-based commands, headers
- Try it yourself:
  - As a "browser" fetching a URL
  - Run "telnet en.wikipedia.org 80"
  - Type "GET /wiki/Vegemite HTTP/1.0" to server followed by a blank line
  - Server will return HTTP response with the page contents (or other info)

# HTTP Protocol (2)

- Commands used in the request

Fetch page → GET

Upload data → POST

| 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 |

# HTTP Protocol (3)

- Codes returned with the response

Yes! →

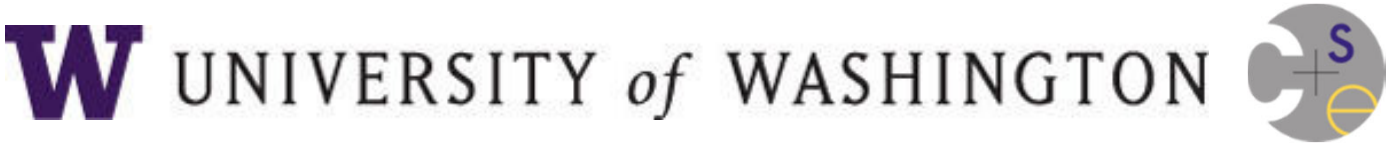
| Code | Meaning | Examples |
|---|---|---|
| 1xx | Information | 100 = server agrees to handle client's request |
| 2xx | Success | 200 = request succeeded; 204 = no content present |
| 3xx | Redirection | 301 = page moved; 304 = cached page still valid |
| 4xx | Client error | 403 = forbidden page;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

# HTTP Protocol (4)

- Many header fields specify capabilities and content
  - E.g., Content-Type: text/html, Cookie: lect=8-4-http

| Function | Example Headers |
|---|---|
| Browser capabilities (client → server) | User-Agent, Accept, Accept-Charset, Accept-Encoding, Accept-Language |
| Caching related (mixed directions) | If-Modified-Since, If-None-Match, Date, Last-Modified, Expires, Cache-Control, ETag |
| Browser context (client → server) | Cookie, Referer, Authorization, Host |
| Content delivery (server → client) | Content-Encoding, Content-Length, Content-Type, Content-Language, Content-Range, Set-Cookie |

# END




© 2013 D. Wetherall

Slide material from: TANENBAUM, ANDREW S.; WETHERALL, DAVID J., COMPUTER NETWORKS, 5th Edition, © 2011. Electronically reproduced by permission of Pearson Education, Inc., Upper Saddle River, New Jersey