

Previously on Computer Networks

- UDP, RTP, RPC
- TCP
 - Connection management
 - Sliding window
 - Timer management
 - Congestion control: Tahoe, Reno, SS/CA, cwnd, threshold

CHAPTER 7

THE APPLICATION LAYER

(应用层)

- **DNS**
- **WWW**
- **EMAIL**
- Multimedia*
- Content Delivery*

DNS (Domain Name System, 域名服务)

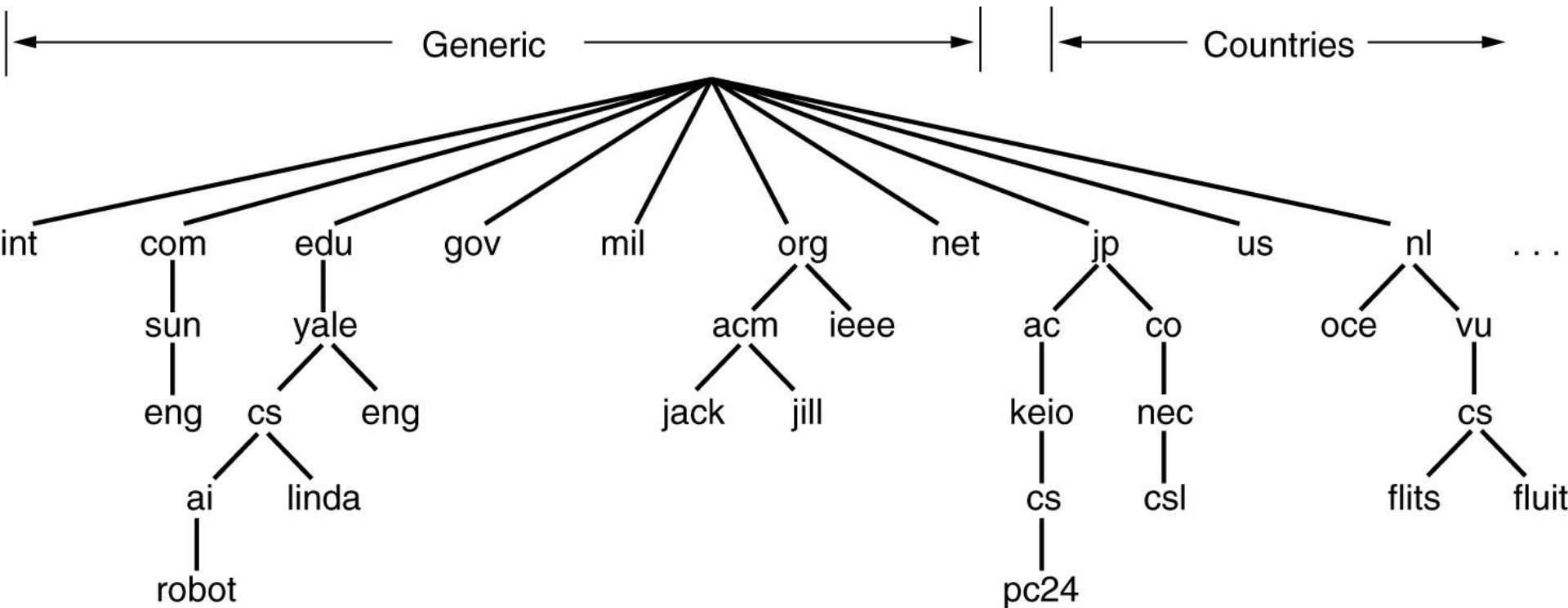
- Why DNS?
- The DNS Name Space
- Resource Records
- Name Servers

DNS: Why?

- Machine addresses vs. machine names
- Hosts.txt
 - Good for a small network
 - The size of the file would be a problem for a large network.
 - Hostnames would conflict.
 - → DNS (Domain Name System).
- DNS (RFCs 1034, 1035, 2181)
 - A hierarchical, domain-based naming scheme
 - A distributed database system for implementing this scheme.

DNS: Name space

A portion of the Internet domain name space.



DNS: Name space

Generic top-level domains

Domain	Intended use	Start date	Restricted?
com	Commercial	1985	No
edu	Educational institutions	1985	Yes
gov	Government	1985	Yes
int	International organizations	1988	Yes
mil	Military	1985	Yes
net	Network providers	1985	No
org	Non-profit organizations	1985	No
aero	Air transport	2001	Yes
biz	Businesses	2001	No
coop	Cooperatives	2001	Yes
info	Informational	2002	No
museum	Museums	2002	Yes
name	People	2002	No
pro	Professionals	2002	Yes
cat	Catalan	2005	Yes
jobs	Employment	2005	Yes
mobi	Mobile devices	2005	Yes
tel	Contact details	2005	Yes
travel	Travel industry	2005	Yes
xxx	Sex industry	2010	No

DNS: Name space

- The top-level domains come in two flavors:
 - generic (*com, edu, gov, int, mil, net, org, biz, info, pro, aero, coop, museum, ...*) and
 - countries (*cn, uk, ...*).
- The lower level-domains are easy to get.
- Each domain is named by the path upward from it to the (unnamed) root, such as *zju.edu.cn*.
- Domain names can be either absolute or relative.
 - An absolute domain name always ends with a period (e.g., *eng.sun.com.*),
 - whereas a relative one does not.

DNS: Name space

- Domain names are **case insensitive**, so *edu*, *Edu*, and *EDU* mean the same thing.
- Component names can be up to 63 characters long, and full path names must not exceed 255 characters.
- Domains can be inserted into the tree in either generic or country domains. For example, sony.com, sony.nl.
- Each domain controls how it allocates the domains under it.
- To create a new domain, permission is required of the domain in which it will be included
- Naming follows organizational boundaries, not physical networks.

DNS: Resource records

- Every domain, whether it is a single host or a top-level domain, can have a set of resource records associated with it.
- A resource record is a five-tuple
(Domain_name, Time_to_live, Class, Type, Value).

The principal DNS resource records types.

Type	Meaning	Value
SOA	Start of authority	Parameters for this zone
A	IPv4 address of a host	32-Bit integer
AAAA	IPv6 address of a host	128-Bit integer
MX	Mail exchange	Priority, domain willing to accept email
NS	Name server	Name of a server for this domain
CNAME	Canonical name	Domain name
PTR	Pointer	Alias for an IP address
SPF	Sender policy framework	Text encoding of mail sending policy
SRV	Service	Host that provides it
TXT	Text	Descriptive ASCII text

A portion of a possible DNS database for *cs.vu.nl*.

; Authoritative data for cs.vu.nl

cs.vu.nl.	86400	IN	SOA	star boss (9527,7200,7200,241920,86400)
cs.vu.nl.	86400	IN	MX	1 zephyr
cs.vu.nl.	86400	IN	MX	2 top
cs.vu.nl.	86400	IN	NS	star

star	86400	IN	A	130.37.56.205
zephyr	86400	IN	A	130.37.20.10
top	86400	IN	A	130.37.20.11
www	86400	IN	CNAME	star.cs.vu.nl
ftp	86400	IN	CNAME	zephyr.cs.vu.nl

flits	86400	IN	A	130.37.16.112
flits	86400	IN	A	192.31.231.165
flits	86400	IN	MX	1 flits
flits	86400	IN	MX	2 zephyr
flits	86400	IN	MX	3 top

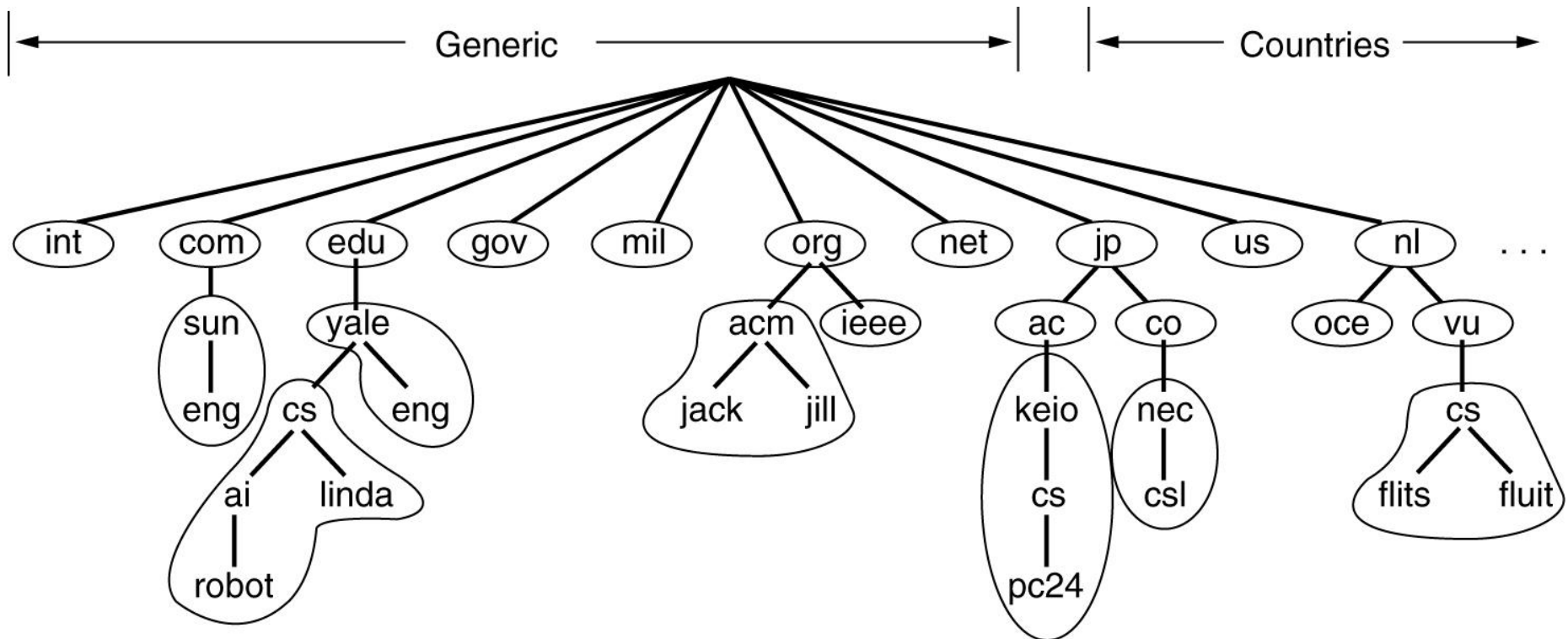
rowboat		IN	A	130.37.56.201
		IN	MX	1 rowboat
		IN	MX	2 zephyr

little-sister		IN	A	130.37.62.23
---------------	--	----	---	--------------

laserjet		IN	A	192.31.231.216
----------	--	----	---	----------------

DNS: Name servers

Part of the DNS name space showing the division into zones.

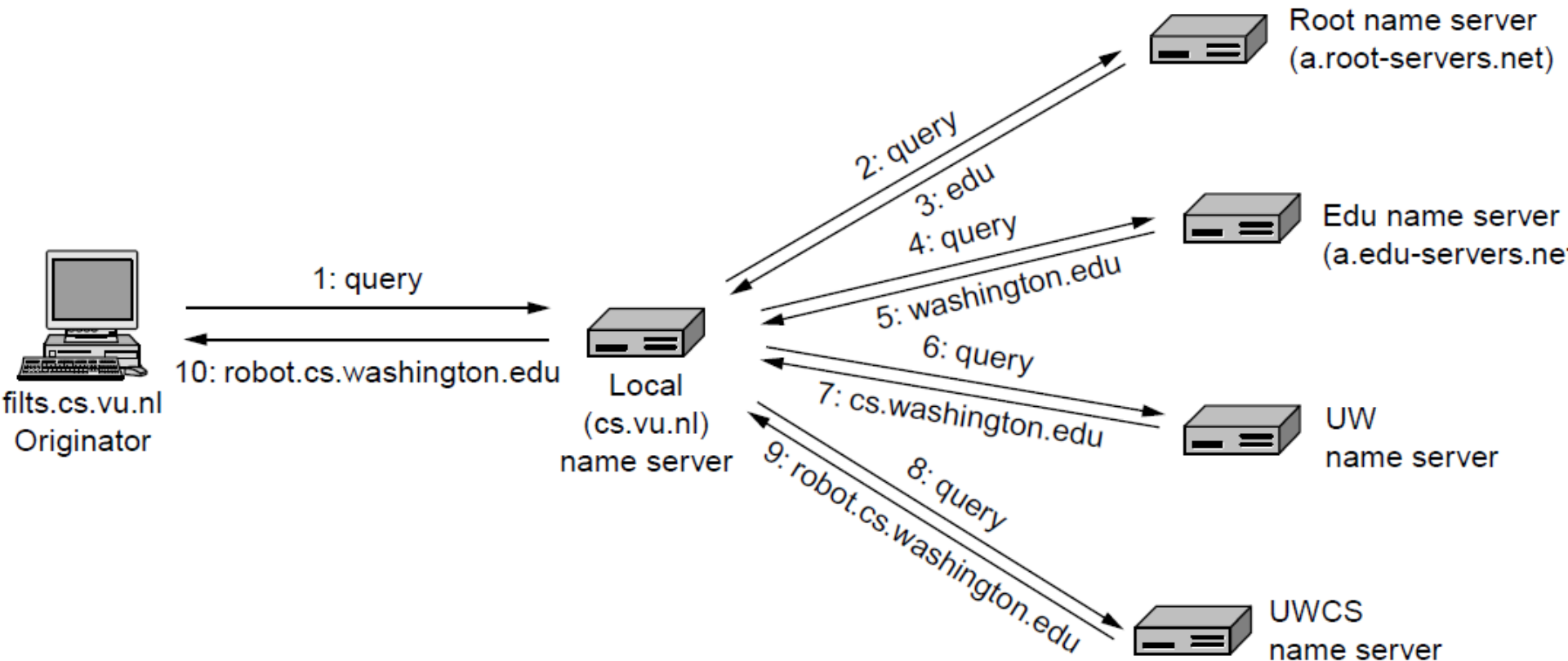


DNS

- An **authoritative record** is one that comes from the authority that manages the record and is thus always correct. → **authoritative server**.
- **Cached records** may be out of date.
- **root name servers:** There are 13 root DNS servers, unimaginatively called *a-root-servers.net* through *m.root-servers.net*. They have information about each top-level domain.

DNS: Name servers

Example of a resolver looking up a remote name **robot.cs.washington.edu** in 10 steps.



DNS: Query mechanism

- **Recursive query:** The mechanism that a host queries the local name server. E.g. when the host *flits.cs.vu.nl* sends its query to the local name server, that name server handles the resolution on behalf of *flits* until it has the desired answer to return. It does *not* return partial answers. They might be helpful, but they are not what the query was seeking.
- **Iterative query.** The mechanism that the local name server queries the root name server and each subsequent name server. e.g. root name server does not recursively continue the query for the local name server. It just returns a partial answer and moves on to the next query. The local name server is responsible for continuing the resolution by issuing further queries.

DNS: useful commands

- **nslookup** -qt=A emnets.org
- **nslookup** -qt=MX mail.emnets.org
- **dig**@a.edu-servers.net robot.cs.washington.edu
// send a query for *robot.cs.washington.edu* to the *a.edu-servers.net* name server and print out the result.

WORLD WIDE WEB

- Architectural Overview
- Static Web Pages
- Dynamic Web Pages
- HTTP – The HyperText Transfer Protocol
- The Mobile Web
- Web Search

WWW: Introduction

- The Web (also known as WWW) began in 1989 at CERN (The European center for nuclear research).
- The initial proposal for a web of linked documents came from CERN physicist **Tim Berners-Lee** in March 1989.
- In December 1991, a public demonstration was given at the Hypertext'91 conference in San Antonio, Texas.
- In 1993 Mosaic was released in February. Mark Andreessen at the University of Illinois.
- In 1995, Netscape Communications Corp went public.
- In 1998, America Online bought Netscape Communication Corp for \$4.2 billion.
- W3C (www.w3.org).

Marc Andreessen

Marc Andreessen is the co-creator of Mosaic, the Web browser that popularized the World Wide Web in 1993. Mosaic had a clean, easily understood interface and was the first browser to display images in-line with text. In 1994, Marc Andreessen and Jim Clark founded Netscape, whose browser was by far the most popular browser through the mid-1990s. Netscape also developed the Secure Sockets Layer (SSL) protocol and many Internet server products, including mail servers and SSL-based Web servers. He is now a co-founder and general partner of venture capital firm Andreessen Horowitz, overseeing portfolio development with holdings that include Facebook, Foursquare, Groupon, Jawbone, Twitter, and Zynga. He serves on numerous boards, including Bump, eBay, Glam Media, Facebook, and Hewlett-Packard. He holds a BS in Computer Science from the University of Illinois at Urbana-Champaign.



Please describe one or two of the most exciting projects you have worked on during your career. What were the biggest challenges?

Undoubtedly the most exciting project was the original Mosaic web browser in '92-'93—and the biggest challenge was getting anyone to take it seriously back then. At the time, everyone thought the interactive future would be delivered as “interactive television” by huge companies, not as the Internet by startups.

Is there anything in particular students should be aware of as Web technology advances?

The rate of change—the most important thing to learn is how to learn—how to flexibly adapt to changes in the specific technologies, and how to keep an open mind on the new opportunities and possibilities as you move through your career.

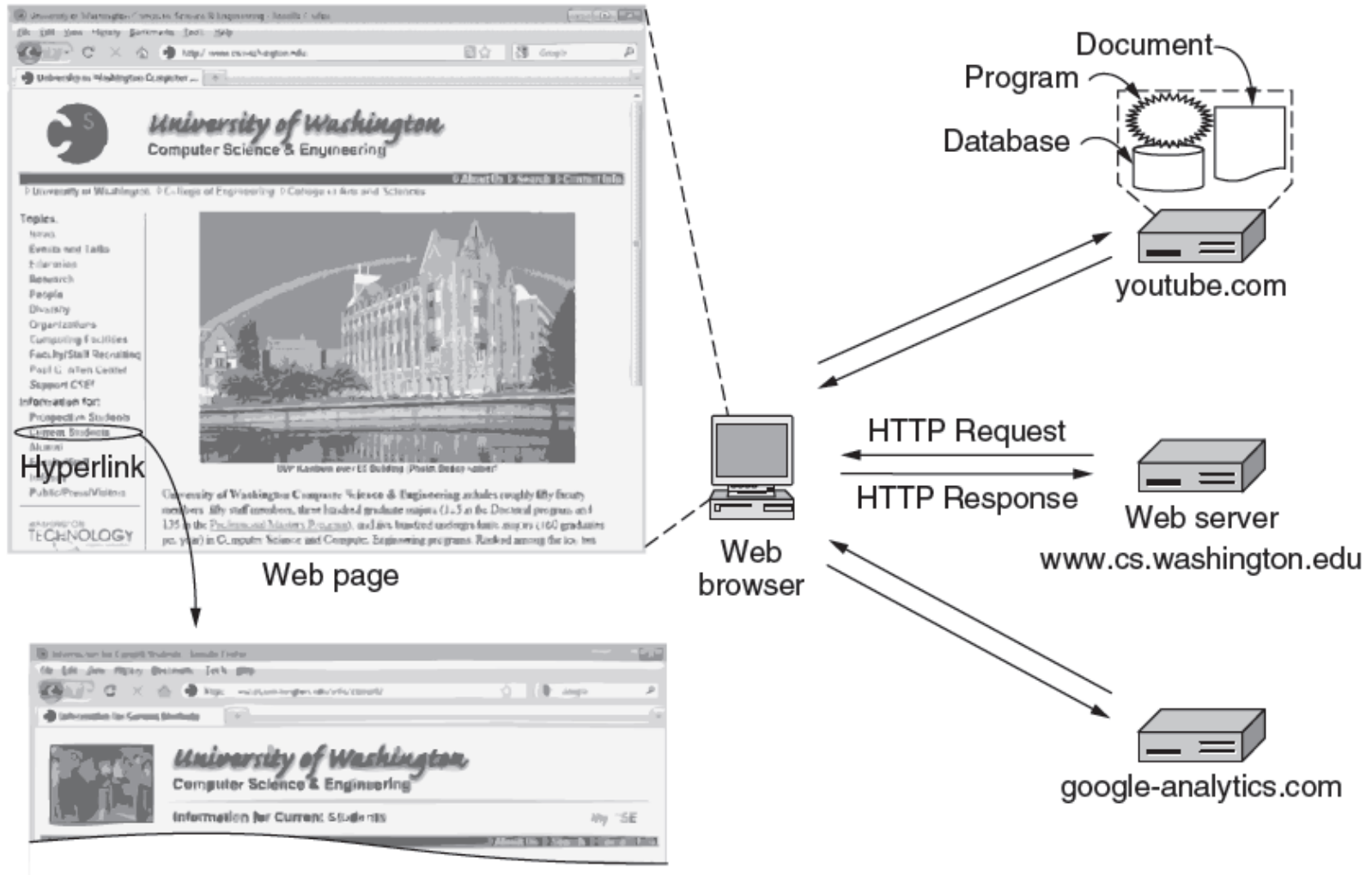
What are your recommendations for students who want to pursue careers in computing and information technology?

Go as deep as you possibly can on understanding how technology is created, and then complement with learning how business works.

Can technology solve the world's problems?

No, but we advance the standard of living of people through economic growth, and most economic growth throughout history has come from technology—so that's as good as it gets.

WWW: Architectural overview



WWW: Architectural overview: Browser

- What is the page called? (Filename)
 - Where is the page located? (DNS Name)
 - How can the page be accessed? (HTTP, FTP, ...)
-
- URL: Universal Resource Locator
 - URL = protocol (http) + DNS name + path name

WWW: URL examples

Name	Used for	Example
http	Hypertext (HTML)	http://www.ee.uwa.edu/~rob/
https	Hypertext with security	https://www.bank.com/accounts/
ftp	FTP	ftp://ftp.cs.vu.nl/pub/minix/README
file	Local file	file:///usr/suzanne/prog.c
mailto	Sending email	mailto:JohnUser@acm.org
rtsp	Streaming media	rtsp://youtube.com/montypython.mpg
sip	Multimedia calls	sip:eve@adversary.com
about	Browser information	about:plugins

WWW: Architectural overview: Browser

What does the browser do when one link is selected:

- The browser determines the URL (by seeing what was selected).
- The browser asks DNS for the IP address of *www.zju.edu.cn*.
- DNS replies with *210.32.0.9*.
- The browser makes a TCP connection to port 80 on *210.32.0.9*.
- It then sends over a request asking for file */home/index.html*.
- The *www.zju.edu.cn* server sends the file */home/index.html*.
- The TCP connection is released.
- The browser displays all the text in */home/index.html*.
- The browser fetches and displays all images in this file.

WWW: URL \rightarrow URI

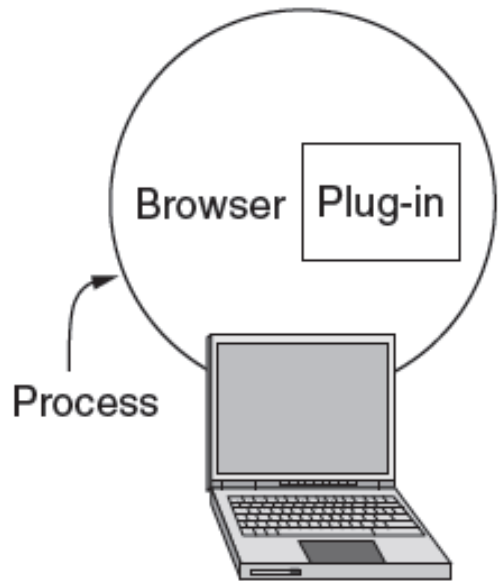
- A URL points to one specific host, but sometimes it is useful to reference a page without simultaneously telling where it is. e.g. “I want page *xyz*, but I do not care where you get it.”
- To solve this kind of problem, **URLs have been generalized into URIs** (Uniform Resource Identifiers).
 - Some URIs tell how to locate a resource. These are the URLs.
 - Other URIs tell the name of a resource but not where to find it. These URIs are called URNs (Uniform Resource Names).
- The rules for writing URIs are given in RFC 3986

WWW: Architectural overview: Browser

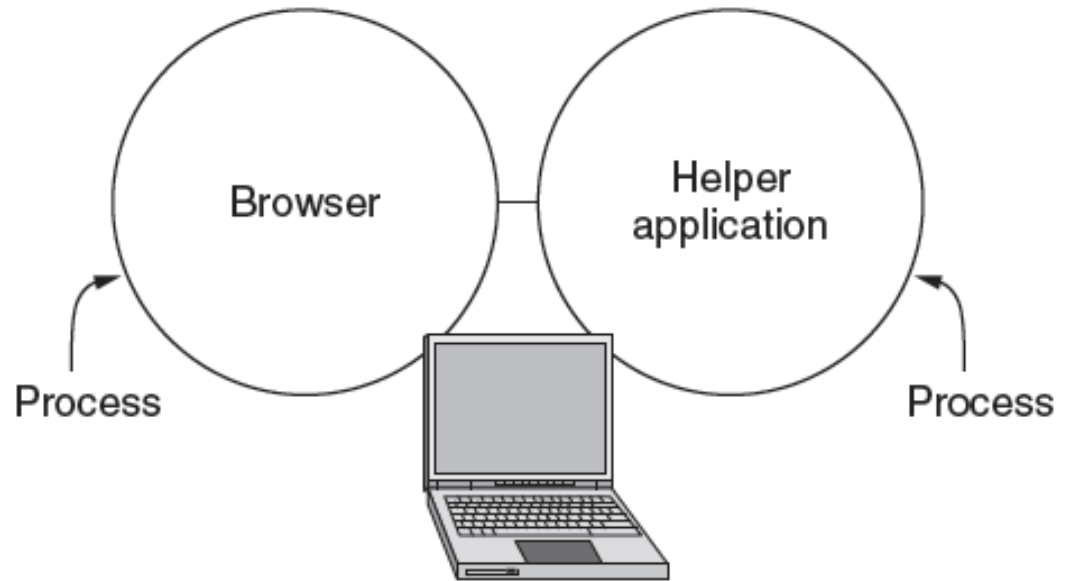
- Browsers can handle many additional types besides HTML pages.
- Browser can use MIME types.
- How to handle other MIME types:
 - **Plug-ins:** A plug-in is a code module that the browser fetches from a special directory on the disk and installs as an extension to itself.
 - **Helper applications:** A help application is a complete program, running as a separate process.

WWW: Architectural overview: Browser

The client side: (a) A browser plug-in. (b) A helper application.



(a)



(b)

WWW: Architectural overview: Server

The browser sends over a command containing the file name on that server. Then the server returns the file for the browser to display.

The short steps that the server performs in its main loop:

1. Accept a TCP connection from a client (a browser).
2. Get the name of the file requested.
3. Get the file (from disk).
4. Return the file to the client.
5. Release the TCP connection.

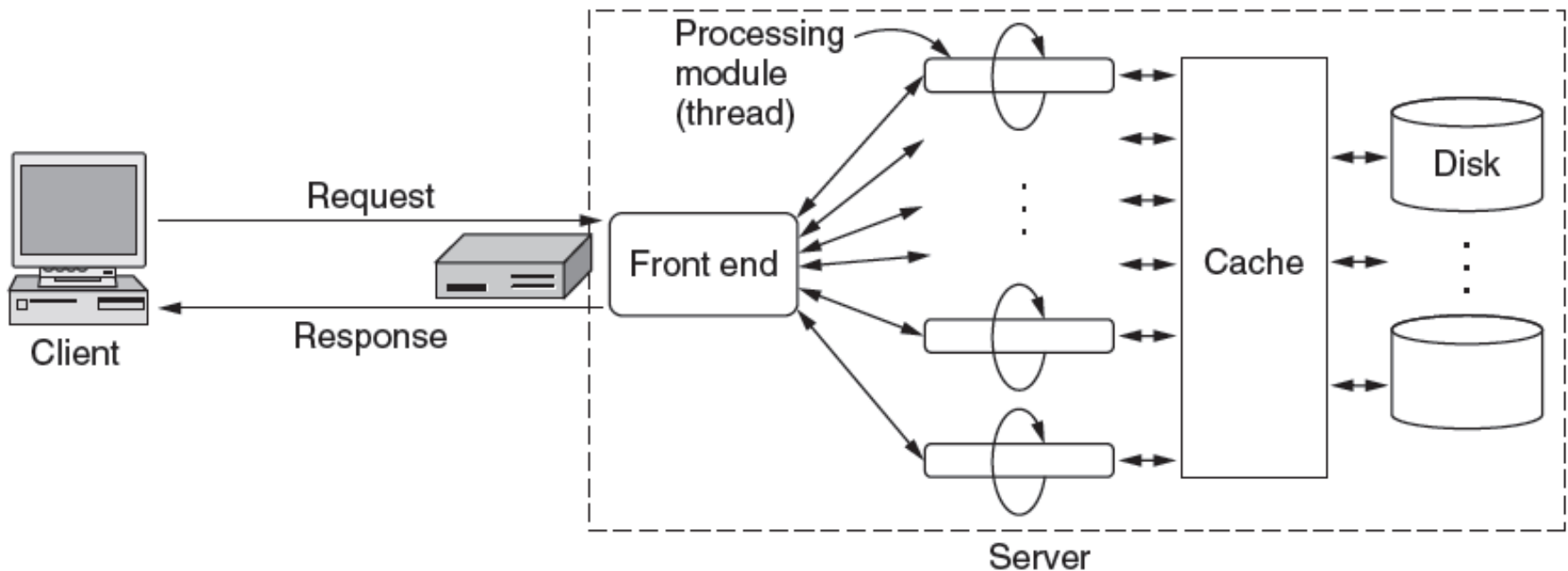
WWW: Architectural overview: Server

The detailed steps that the server performs in its main loop are:

1. Resolve the name of the Web page requested.
2. Perform access control on the Web page.
3. Check the cache.
4. Fetch the requested page from disk or run a program to build it.
5. Determine the rest of the response (e.g., the MIME type to send).
6. Return the response to the client.
7. Make an entry in the server log.

WWW: Architectural overview: Server

A multithreaded Web server with a front end and processing modules.



WWW: Architectural overview: Cookies

- The Web is basically stateless.
 - How to distinguish between requests from registered users and everyone else?
 - How to keep track of the contents of the cart (for e-commerce)?
 - How to customize Web portals such as Yahoo?
- To track users by observing their IP addresses?
 - The IP address merely identifies the computers, not the user.
 - Many ISP use NAT (Network Address Translation).
- To track users by cookies.

WWW: Architectural overview: Cookies

- A cookie may contain up to 5 fields
 - **Domain:** where the cookie came from
 - **Path:** which parts of the server's file tree may use it
 - **Content:** *name=value* lists
 - **Expires:** when the cookie expires.
 - **Secure:** can be set to indicate that the browser may only return the cookie to a server using a secure transport.

Domain	Path	Content	Expires	Secure
toms-casino.com	/	CustomerID=297793521	15-10-10 17:00	Yes
jills-store.com	/	Cart=1-00501;1-07031;2-13721	11-1-11 14:22	No
aportal.com	/	Prefs=Stk:CSCO+ORCL;Spt:Jets	31-12-20 23:59	No
sneaky.com	/	UserID=4627239101	31-12-19 23:59	No

WWW: Architectural overview: Cookies

- The first cookie was set by toms-casino.com and is used to identify the customer.
- When the client returns next week to throw away some more money, the browser sends over the cookie so the server knows who it is.
- Armed with the customer ID, the server can look up the customer's record in a database and use this information to build an appropriate Web page to display.
- Depending on the customer's known gambling habits, this page might consist of a poker hand, a listing of today's horse races, or a slot machine.

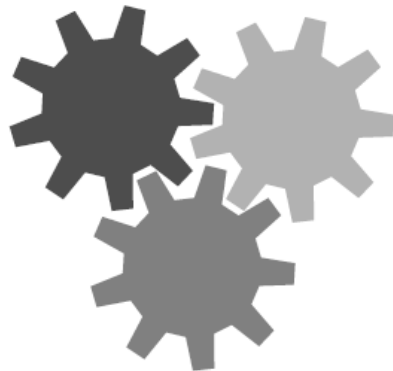
WWW: Static Web Pages: HTML

- HTML (HyperText Markup Language)
 - (Markup for copyeditors → troff → Tex/Latex → HTML)
- A web page consists of a head and a body, each enclosed by `<html>` and `</html>`
 - The head is bracketed by the `<head>` and `</head>`
 - The body is bracketed by the `<body>` and `</body>`
- Tags:
 - `<something>` and `</something>`
 - Can have attributes
 - Can be in either lower case or upper case, but lower case is best for compatibility.
- Some special characters such as ` `; (a space)

WWW: Static Web Pages: HTML example

```
<html>
<head><title> AMALGAMATED  WIDGET, INC. </title> </head>
<body> <h1> Welcome to AWI's Home Page</h1>
 <br>
We are so happy that you have chosen to visit <b> Amalgamated Widget's </b>
home page. We hope <i> you </i> will find all the information you need here.
<p>Below we have links to information about our many fine products.
You can order electronically (by WWW), by telephone, or by fax. </p>
<hr>
```

Welcome to AWI's Home Page



We are so happy that you have chosen to visit **Amalgamated Widget's** home page. We hope *you* will find all the information you need here.

Below we have links to information about our many fine products. You can order electronically (by WWW), by telephone, or by email.

WWW: Static Web Pages: HTML example

```
<h2> Product information </h2>
<ul>
  <li> <a href="http://widget.com/products/big"> Big widgets </a>
  <li> <a href="http://widget.com/products/little"> Little widgets </a>
</ul>
<h2> Telephone numbers</h2>
<ul>
  <li> By telephone: 1-800-WIDGETS
  <li> By fax: 1-415-765-4321
</ul>
</body>
</html>
```

Product Information

- [Big widgets](#)
- [Little widgets](#)

Contact information

- By telephone: 1-800-WIDGETS
- By email: info@amalgamated-widget.com

WWW: Static Web Pages: HTML:

A selection of common HTML tags. some can have additional parameters.

Tag	Description
<html> ... </html>	Declares the Web page to be written in HTML
<head> ... </head>	Delimits the page's head
<title> ... </title>	Defines the title (not displayed on the page)
<body> ... </body>	Delimits the page's body
<h <i>n</i> > ... </h <i>n</i> >	Delimits a level <i>n</i> heading
 ... 	Set ... in boldface
<i> ... </i>	Set ... in italics
<center> ... </center>	Center ... on the page horizontally
 ... 	Brackets an unordered (bulleted) list
 ... 	Brackets a numbered list
	Starts a list item (there is no)
 	Forces a line break here
<p>	Starts a paragraph
<hr>	Inserts a Horizontal rule
	Displays an image here
 ... 	Defines a hyperlink

WWW: Static Web Pages: HTML versions

Item	HTML 1.0	HTML 2.0	HTML 3.0	HTML 4.0	HTML 5.0
Hyperlinks	x	x	x	x	x
Images	x	x	x	x	x
Lists	x	x	x	x	x
Active maps & images		x	x	x	x
Forms		x	x	x	x
Equations			x	x	x
Toolbars			x	x	x
Tables			x	x	x
Accessibility features				x	x
Object embedding				x	x
Style sheets				x	x
Scripting				x	x
Video and audio					x
Inline vector graphics					x
XML representation					x
Background threads					x
Browser storage					x
Drawing canvas					x

WWW: Static Web Pages: Form

- `<form> </form>`
 - `<input name="variable_name"`
 `type="text|textarea|password|radio|check|submit"`
 `size="a_value" >`
 - `<form action="" method="POST|GET">`

WWW: Static Web Pages: Form

```
<html>
<head> <title> AWI CUSTOMER ORDERING FORM </title> </head>
<body>
<h1> Widget Order Form </h1>
<form ACTION="http://widget.com/cgi-bin/widgetorder" method=POST>
<p> Name <input name="customer" size=46> </p>
<p> Street Address <input name="address" size=40> </p>
<p> City <input name="city" size=20> State <input name="state" size =4>
Country <input name="country" size=10> </p>
```

Widget Order Form

Name

Street address

City

State

Country

WWW: Static Web Pages: Form

```
<p> Credit card # <input name="cardno" size=10>
Expires <input name="expires" size=4>
M/C <input name="cc" type=radio value="mastercard">
VISA <input name="cc" type=radio value="visacard"> </p>
<p> Widget size Big <input name="product" type=radio value="expensive">
Little <input name="product" type=radio value="cheap">
Ship by express courier <input name="express" type=checkbox> </p>
<p><input type=submit value="submit order"> </p>
Thank you for ordering an AWI widget, the best widget money can buy!
</form>
</body>
</html>
```

Credit card # Expires M/C ☐ Visa ☐

Widget size Big ☐ Little ☐ Ship by express courier ☐

Thank you for ordering an AWI widget, the best widget money can buy!

WWW: Static Web Pages: Form

- When the user clicks the submit button, the browser packages the collected information into a single long line time and sends it back to the server for processing.

```
customer=John+Doe&address=100+Main+St.&city=White+Plains&  
state=NY&country=USA&cardno=1234567890&expires=6/98&cc=mastercard&  
product=cheap&express=on
```

WWW: Static Web Pages: CSS

- The original goal of HTML
 - structure
 - Style
- Some styles

`<font face="helvetica"`

`size="24"`

`color="red"> Deborah's Photos`

``

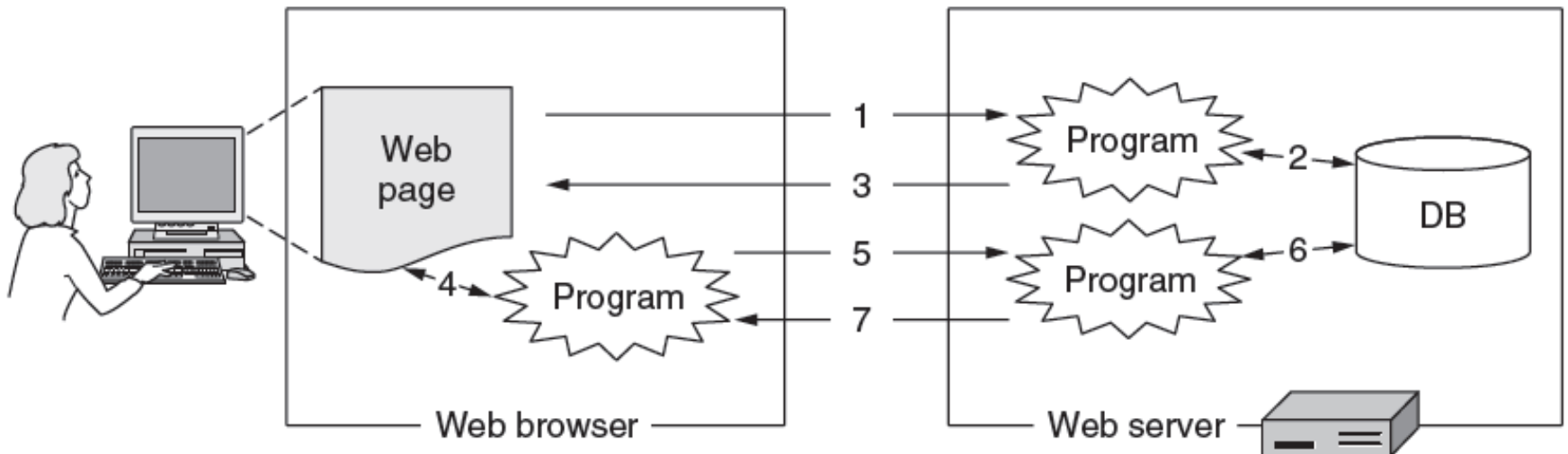
WWW: Static Web Pages: CSS

- Defining a **CSS** (Cascading Style Sheets) style sheet

```
body {background-color:linen; color:navy; font-family:Arial;}
h1 {font-size:200%;}
h2 {font-size:150%;}
```
- Using a CSS style sheet

```
<head>
<title> AMALGAMATED WIDGET, INC. </title>
<link rel="stylesheet" type="text/css" href="awistyle.css" />
</head>
```

WWW: Dynamic Web Pages:



WWW: Dynamic Web Pages:

Server-side dynamic web page generation

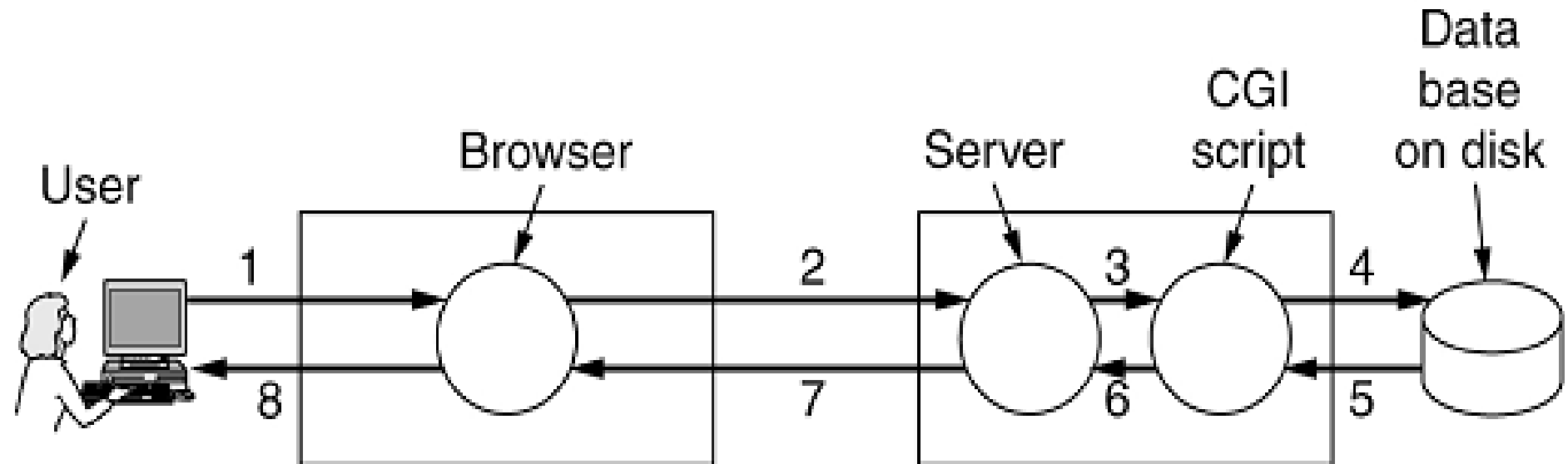
- Two methods
 - **CGI (Common Gateway Interface)**: provides an interface to allow Web servers to talk to back-end programs and scripts (e.g. **Python, Ruby, Perl**) that can accept input (e.g. from forms) and generate HTML pages in response.
 - **Server pages**: to embed little scripts inside HTML pages and have them be executed by the server itself to generate the page.
 - **PHP**: Hypertext Preprocessor
 - **JSP**: Java Server Pages
 - **ASP.NET**: Active Server Page

WWW: Dynamic Web Pages:

Server-side dynamic web page generation: CGI

Steps in processing the information from an HTML form

1. User fills in form
2. Form sent back
3. Handed to CGI
4. CGI queries DB
5. Record found
6. CGI builds page
7. Page returned
8. Page displayed



WWW: Dynamic Web Pages:

Server-side dynamic web page generation:

PHP, JSP, ASP/ASP.NET

- **PHP (Hypertext Preprocessor)**
- **JSP (JavaServer Pages):** the dynamic part is written in the Java programming language. Pages using this technique have the file extension .jsp.
- **ASP.NET (Active Server Pages .NET)** is Microsoft's version of PHP and JavaServer Pages. It uses programs written in Microsoft's proprietary .NET networked application framework for generating the dynamic content. Pages using this technique have the extension .aspx.

WWW: Dynamic Web Pages:

Server-side dynamic web page generation: PHP

```
<html>
<body>
<form action="action.php" method="post">
<p> Please enter your name: <input type="text" name="name"> </p>
<p> Please enter your age: <input type="text" name="age"> </p>
<input type="submit">
</form>
</body>
</html>
```

- (a) A Web page containing a form. the <form> tag specifies that *action.php* is to be invoked to handle the parameters when the user submits the form

WWW: Dynamic Web Pages:

Server-side dynamic web page generation: PHP

```
<html>
<body>
<h1> Reply: </h1>
Hello <?php echo $name; ?>.
Prediction: next year you will be <?php echo $age + 1; ?>
</body>
</html>
```

- (b) A PHP script for handling the output of the form. The server then starts to process the action.php file as a reply

WWW: Dynamic Web Pages:

Server-side dynamic web page generation: PHP

```
<html>  
<body>  
<h1> Reply: </h1>  
Hello Barbara.  
Prediction: next year you will be 33  
</body>  
</html>
```

(c) Output from the PHP script when the inputs are “Barbara” and “32”, respectively.

WWW: Dynamic Web Pages:

Client-side dynamic web page generation

- Server-side dynamic web page generation solve the problem of handling input and interactions with databases on the server. They can all accept incoming information from forms, look up information in one or more databases, and generate HTML pages with the results.
- **What none of them can do is respond to mouse movements or interact with users directly.** For this purpose, it is necessary to have scripts embedded in HTML pages that are executed on the client machine rather than the server machine.
- Starting with HTML 4.0, these scripts are permitted using the tag <script> and the technologies used to produce these interactive Web pages are broadly referred to as **dynamic HTML**

WWW: Dynamic Web Pages:

Client-side dynamic web page generation

- The most popular scripting language for the client side is **JavaScript**.
- JavaScript has almost nothing to do with the Java programming language.
- It is mutating very fast and thus is extremely difficult to write JavaScript programs that work on all platforms, but maybe some day it will stabilize.

WWW: Dynamic Web Pages:

Client-side dynamic web page generation

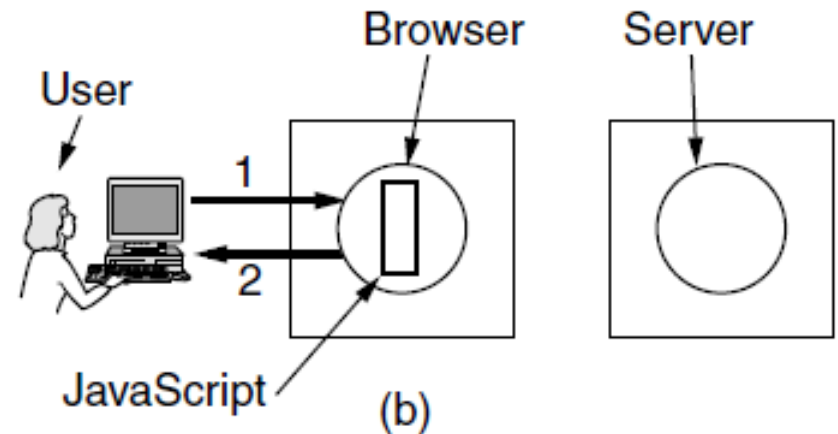
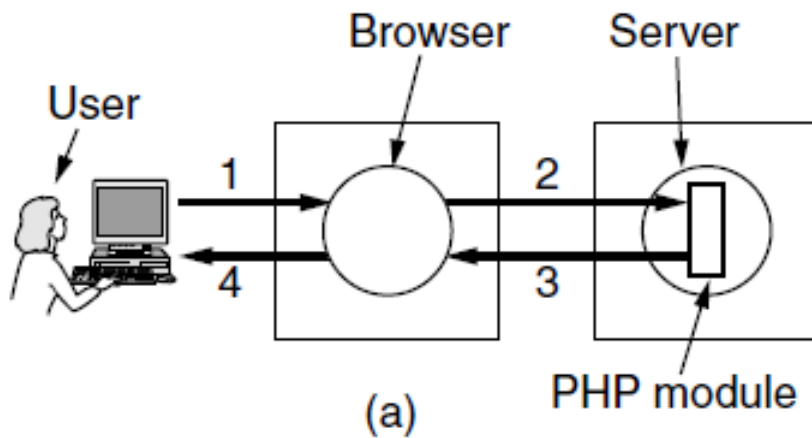
```
<html>
<head>
<script language="javascript" type="text/javascript">
function response(test_form) {
    var person = test_form.name.value;
    var years = eval(test_form.age.value) + 1;
    document.open();
    document.writeln("<html> <body>");
    document.writeln("Hello " + person + ".<br>");
    document.writeln("Prediction: next year you will be " + years + ".");
    document.writeln("</body> </html>");
    document.close();
}
</script>
</head>
```

```
<body>
<form>
Please enter your name: <input type="text" name="name">
<p>
Please enter your age: <input type="text" name="age">
<p>
<input type="button" value="submit" onclick="response(this.form)">
</form>
</body>
</html>
```

WWW: Dynamic Web Pages: server-side scripting vs client-side scripting

(a) Server-side scripting with PHP.

(b) Client-side scripting with JavaScript.



WWW: Dynamic Web Pages:

Client-side dynamic web page generation

- **JavaScript** is very popular way to make Web pages highly interactive.
- **VBScript** (based on Visual Basic) is an alternative on Windows platforms.
- **Java applets** runs on the JVM (Java Virtual Machine)
- **ActiveX Controls** (which are programs compiled to x86 machine language and executed on the bare hardware) from Microsoft runs on Windows.

WWW: Dynamic Web Pages:

AJAX (Asynchronous JavaScript And XML)

- **AJAX** is a set of technologies that work together to enable Web applications that are every bit as responsive and powerful as traditional desktop applications. The technologies are:
 1. **HTML and CSS** to present information as pages.
 2. **XML (eXtensible Markup Language)** to let programs exchange application data with the server.
 3. **DOM (Document Object Model)** to change parts of pages while they are viewed.
 4. **An asynchronous way** for programs to send and retrieve XML data.
 5. **JavaScript** as a language to bind all this functionality together.

WWW: Dynamic Web Pages:

AJAX (Asynchronous JavaScript And XML): XML

- **XML** (eXtensible Markup Language), is a language for specifying structured content.
- HTML mixes content with formatting because it is concerned with the presentation of information.
- XML allows Web content to be structured for automated processing.
- There are no defined tags for XML. Each user can define her own tags.

WWW: Dynamic Web Pages:

AJAX (Asynchronous JavaScript And XML): XML

```
<?xml version="1.0" ?>
```

```
<book_list>
```

```
<book>
```

```
<title> Human Behavior and the Principle of Least Effort </title>
```

```
<author> George Zipf </author>
```

```
<year> 1949 </year>
```

```
</book>
```

```
<book>
```

```
<title> The Mathematical Theory of Communication </title>
```

```
<author> Claude E. Shannon </author>
```

```
<author> Warren Weaver </author>
```

```
<year> 1949 </year>
```

```
</book>
```

```
<book>
```

```
<title> Nineteen Eighty-Four </title>
```

```
<author> George Orwell </author>
```

```
<year> 1949 </year>
```

```
</book>
```

```
</book_list>
```

WWW: Dynamic Web Pages:

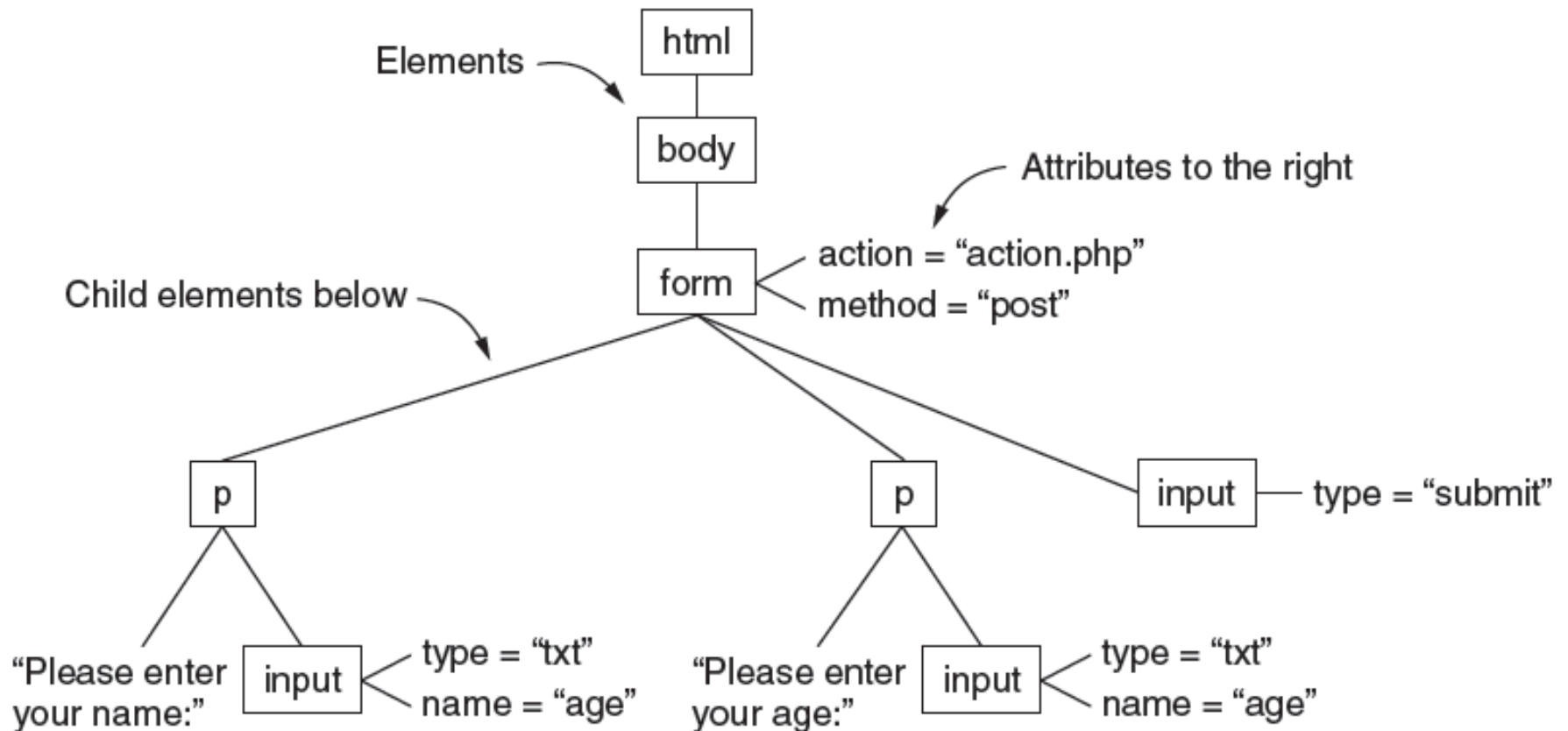
AJAX (Asynchronous JavaScript And XML): XML

- **XSLT (eXtensible Stylesheet Language Transformations)** can be used to define how XML should be transformed into HTML. XSLT is like CSS, but much more powerful.
- **The data expressed in XML**, instead of HTML, is that it is easier for programs to analyze.

WWW: Dynamic Web Pages:

AJAX (Asynchronous JavaScript And XML): DOM

A DOM tree example



WWW: Dynamic Web Pages:

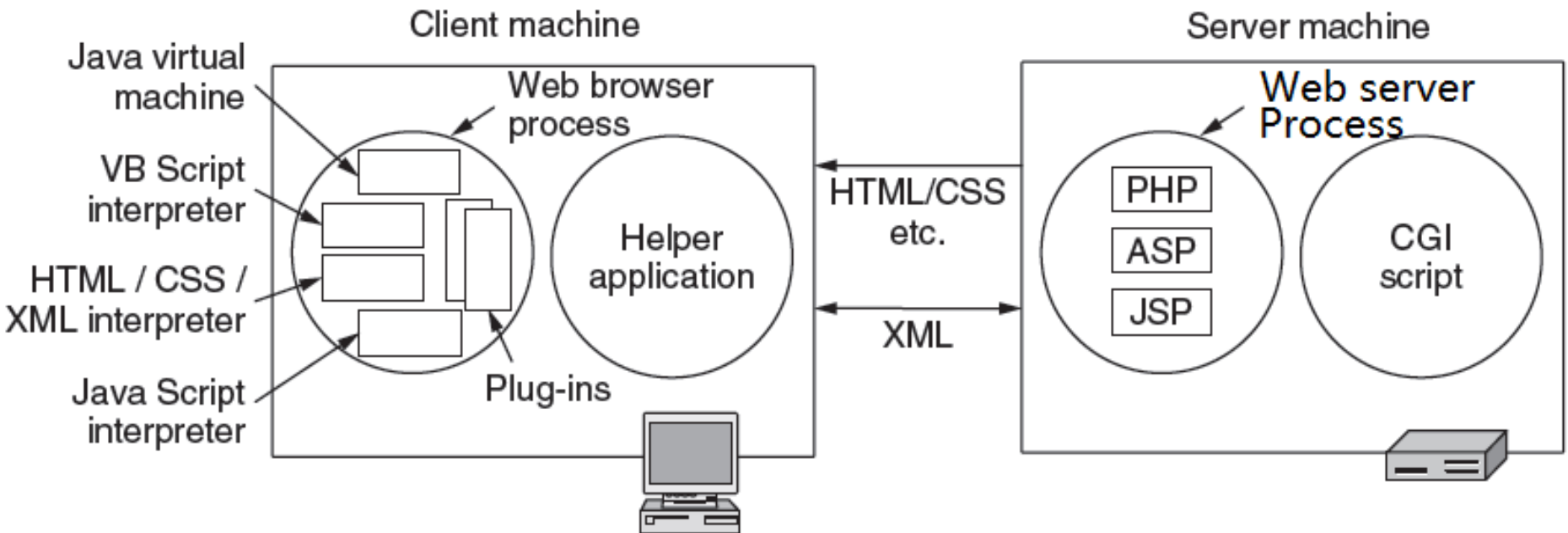
AJAX (Asynchronous JavaScript And XML): A-IO

To provide a responsive interface in the browser while sending or receiving data, it must be possible for scripts to perform **asynchronous I/O** that does not block the display while awaiting the response to a request.

- For example, consider a map that can be scrolled in the browser. When it is notified of the scroll action, the script on the map page may request more map data from the server if the view of the map is near the edge of the data.
- The interface should not freeze while those data are fetched.
- Instead, the scrolling should continue smoothly.
- When the data arrive, the script is notified so that it can use the data. If all goes well, new map data will be fetched before it is needed.
- **Modern browsers have support for this model of communication.**

WWW: Dynamic Web Pages: Summary

The various ways to generate and display content.



WWW: HTTP (HyperText Transfer Protocol)

- **HTTP** is an application layer protocol because it runs on top of TCP and is closely associated with the Web.
- Now HTTP is becoming more like a transport protocol that provides a way for processes to communicate content across the boundaries of different networks. These processes do not have to be a Web browser and Web server.
 - A media player could use HTTP to talk to a server and request album information.
 - Antivirus software could use HTTP to download the latest updates.

WWW: HTTP (HyperText Transfer Protocol)

- Various usages of HTTP
 - Developers could use HTTP to fetch project files.
 - Consumer electronics products like digital photo frames often use an embedded HTTP server as an interface to the outside world.
 - Machine-to-machine communication increasingly runs over HTTP. For example, an airline server might use
 - SOAP (an XML RPC over HTTP) to contact a car rental server and make a car reservation, all as part of a vacation package. These trends are likely to continue, along with the expanding use of HTTP.

WWW: HTTP

- Connections
- Methods
- Message Headers (parameters)
 - Request headers
 - Response headers
- Caching
- Experimenting with HTTP
 - telnet www.ietf.org 80

WWW: HTTP: Connections

- HTTP 1.0 with multiple connections
 - Establish a TCP connection
 - Send a request
 - Get a single response
 - Terminate the TCP connection
- HTTP 1.1 with persistent connections
 - Establish a TCP connection
 - Looping for sending a request and getting a single response
 - Terminate the TCP connection
- HTTP 1.1 with persistent connections and pipeline requests.

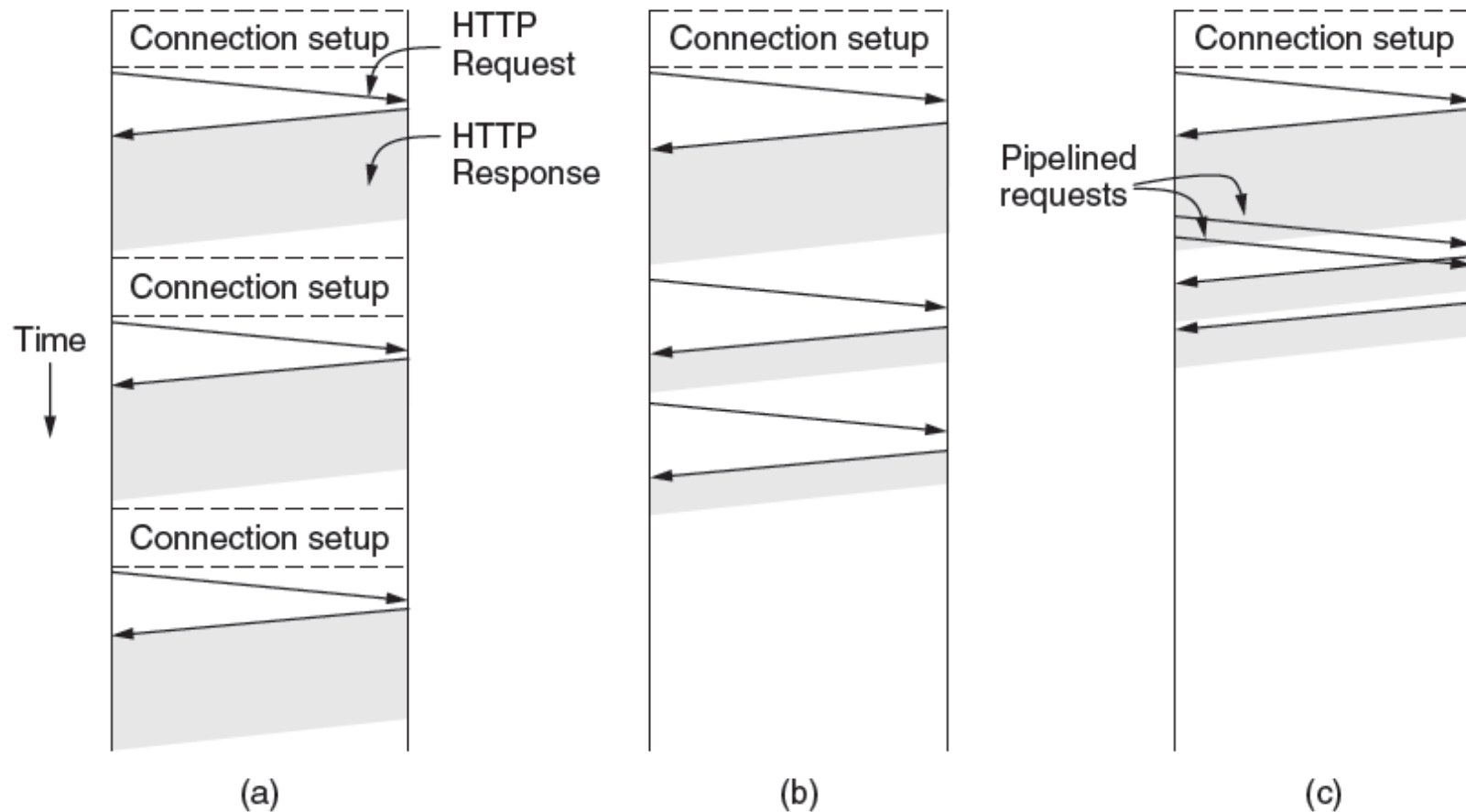
WWW: HTTP: Connections

HTTP with

(a) multiple connections and sequential requests.

(b) A persistent connection and sequential requests.

(c) A persistent connection and pipelined requests.



Some definitions

- **RTT**: round trip time; is the time it takes for a small packet to travel from client to server and then back to the client.
- The RTT includes
 - packet propagation delays,
 - packet queuing delays in intermediate routers and switches
 - packet processing delays.
- also known as the **ping time**

- Suppose within your Web browser you click on a link to obtain a Web page. Suppose that the IP address for the associated URL is not cached in your local host, so that a DNS lookup is necessary to obtain the IP address. Suppose that n DNS servers are visited before your host receives the IP address from DNS; the successive visits incur an RTT of RTT_1, \dots, RTT_n . Further suppose that the Web page associated with the link contains exactly one object, a small amount of HTML text. Let RTT_0 denote the RTT between the local host and the server containing the object. Assuming *zero* transmission time of the object, how much time elapses from when the client clicks on the link until the client receives the object?
- Suppose the HTML file indexes *three* very small objects on the same server. Neglecting transmission times, how much time elapses with
 - (1) Multiple HTTP connections and sequential requests,
 - (2) persistent HTTP connection and sequential requests,
 - (3) persistent HTTP connection and pipelined requests

WWW: HTTP: Methods

The built-in HTTP request methods.

Method	Description
GET	Request to read a Web page
HEAD	Request to read a Web page's header
PUT	Request to store a Web page
POST	Append to a named resource (e.g., a Web page)
DELETE	Remove the Web page
TRACE	Echo the incoming request
CONNECT	Reserved for future use
OPTIONS	Query certain options

WWW: HTTP: Methods

Each request consists of one or more lines of ASCII text, with the first word on the first line being the name of the method requested. The method names are **case sensitive**, so *GET* is allowed but not *get*.

- The **GET** method requests the server to send the page.
- The **HEAD** method just asks for the message header, without the actual page. This method can be used to collect information for indexing purposes, or just to test a URL for validity.
- The **POST** method is used when forms are submitted. Both it and GET are also used for SOAP Web services.
 - It uploads data to the server (i.e., the contents of the form or RPC parameters).
 - The server then does something with the data that depends on the URL, conceptually appending the data to the object.
 - Finally, the method returns a page indicating the result.

WWW: HTTP: Methods

- The **PUT** method is the reverse of GET: instead of reading the page, it writes the page. This method makes it possible to build a collection of Web pages on a remote server.
- **DELETE** removes the page, or at least it indicates that the Web server has agreed to remove the page. As with PUT, authentication and permission play a major role here.
- The **TRACE** method is for debugging. It instructs the server to send back the request.
- The **CONNECT** method lets a user make a connection to a Web server through an intermediate device, such as a Web cache.
- The **OPTIONS** method provides a way for the client to query the server for a page and obtain the methods and headers that can be used with that page.

WWW: HTTP: Methods

The status code response groups.

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

WWW: HTTP: Message headers

- The request line (e.g., the line with the *GET* method) may be followed by additional lines with more information.
- This information (**request headers**) can be compared to the parameters of a procedure call.
- Responses may also have **response headers**.
- Some headers can be used in either direction.

WWW: HTTP: Message 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
Referer	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

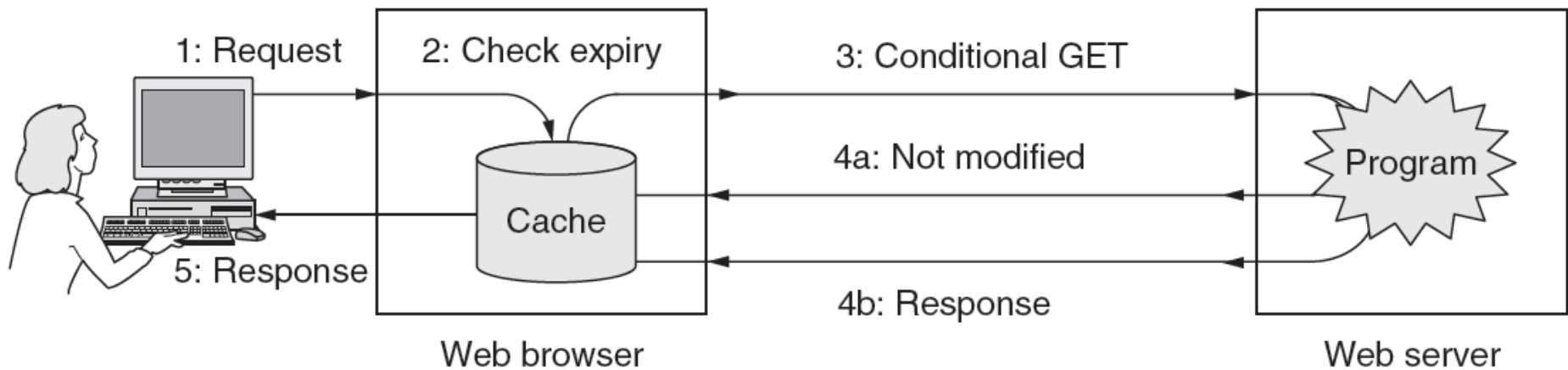
WWW: HTTP: Message headers

Content-Encoding	Response	How the content is encoded (e.g., <i>gzip</i>)
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

WWW: HTTP: Caching

- HTTP has built-in *caching* support.
- How to determine that a previously cached copy of a page is the same as the page would be if it was fetched again?
 - The first strategy is *page validation* (step 2). The cache is consulted, and if it has a copy of a page for the requested URL that is known to be fresh (i.e., still valid), there is no need to fetch it anew from the server.
 - The second strategy is to ask the server if the cached copy is still valid. This request is a *conditional GET* (step 3).
 - Both of these caching strategies are overridden by the directives carried in the Cache-Control header (e.g. no-cache).
- Other caching issues: proxy caching, caching effect,

WWW: HTTP: Caching



WWW: HTTP: Experimenting with HTTP

- In most UNIX shells and the command window on Windows (once the telnet program is enabled).

```
telnet www.ietf.org 80
```

```
GET /rfc.html HTTP/1.1
```

```
Host: www.ietf.org
```

```
# Comment line:
```

```
#A blank line following the last header is mandatory.
```

WWW: HTTP: An Example

```
Trying 4.17.168.6...
Connected to www.ietf.org.
Escape character is '^]'.
HTTP/1.1 200 OK
Date: Wed, 08 May 2002 22:54:22 GMT
Server: Apache/1.3.20 (Unix) mod_ssl/2.8.4 OpenSSL/0.9.5a
Last-Modified: Mon, 11 Sep 2000 13:56:29 GMT
ETag: "2a79d-c8b-39bce48d"
Accept-Ranges: bytes
Content-Length: 3211
Content-Type: text/html
X-Pad: avoid browser bug
```

The start of the output of
www.ietf.org/rfc.html.

telnet www.ietf.org 80>log

GET /rfc.html HTTP/1.1

HOST:www.ietf.org

close

```
<html>
<head>
<title>IETF RFC Page</title>

<script language="javascript">
function url() {
  var x = document.form1.number.value
  if (x.length == 1) {x = "000" + x }
  if (x.length == 2) {x = "00" + x }
  if (x.length == 3) {x = "0" + x }
  document.form1.action = "/rfc/rfc" + x + ".txt"
  document.form1.submit
}
</script>

</head>
```


WWW: The Mobile Web

Difficulties for mobile phones browsing the web

1. **Relatively small screens** preclude large pages and large images.
2. **Limited input capabilities** make it tedious to enter URLs or other lengthy input.
3. **Network bandwidth** is limited over wireless links, particularly on cellular (3G) networks, where it is often expensive too.
4. **Connectivity** may be intermittent.
5. **Computing power** is limited, for reasons of battery life, size, heat dissipation, and cost.

WWW: The Mobile Web

- **WAP** (Wireless Application Protocol): failed
- Increasingly used approach: run the same Web protocols for mobiles and desktops, and to have Web sites deliver *mobile-friendly* content when mobile
 - W3C is encouraging this approach in several ways. E.g. standardize best practices for mobile Web content
- **XHTML basic**: stripped-down version of HTML
- a complementary approach is the use of *content transformation* or *transcoding*. a computer sits between the mobile and the server, transforms pages to mobile friendly, e.g. reduce image size.

WWW: The Mobile Web: XHTML Basic

The XHTML Basic modules and tags.

Module	Req.?	Function	Example tags
Structure	Yes	Doc. structure	body, head, html, title
Text	Yes	Information	br, code, dfn, em, <i>h<i>n</i></i> , kbd, p, strong
Hypertext	Yes	Hyperlinks	a
List	Yes	Itemized lists	dl, dt, dd, ol, ul, li
Forms	No	Fill-in forms	form, input, label, option, textarea
Tables	No	Rectangular tables	caption, table, td, th, tr
Image	No	Pictures	img
Object	No	Applets, maps, etc.	object, param
Meta-information	No	Extra info	meta
Link	No	Similar to <a>	link
Base	No	URL starting point	base

WWW: Web Search

- Web search companies
 - Google, Bing
 - Baidu
- How does web search works?
 - How to find web pages? **Web crawling**
 - How to store web pages? **Hard disks**
 - How to process web pages? **Indexing techniques**
 - How to make profit from web search? **Advertising**
 -

Email

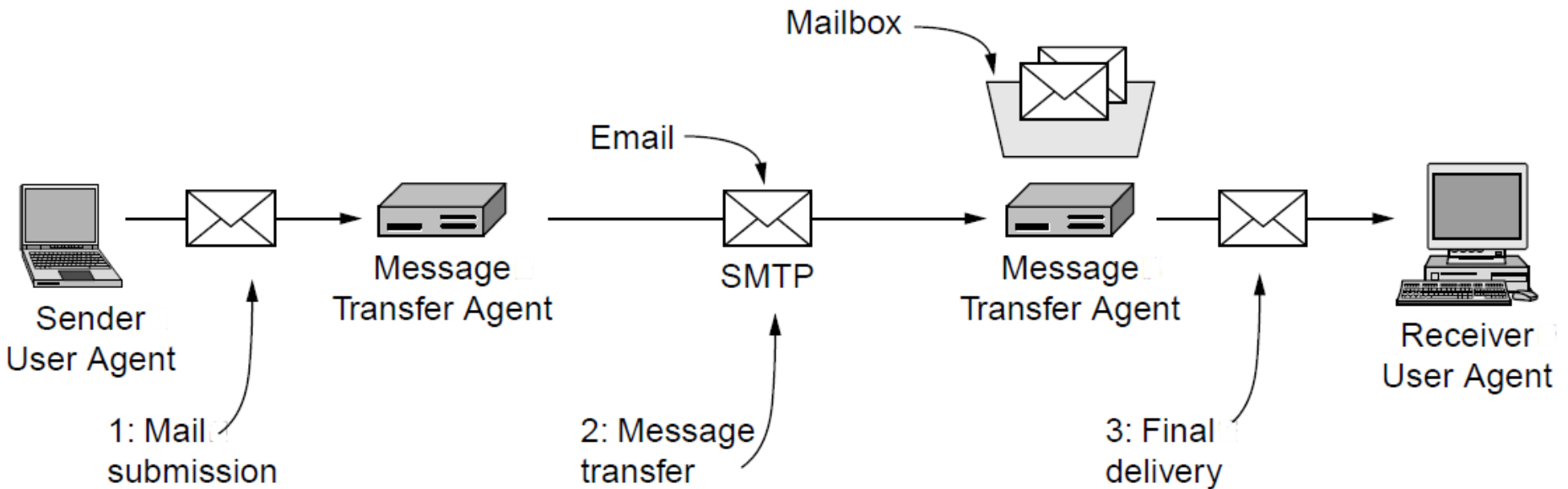
- Architecture and Services
- The User Agent
- Message Formats
- Message Transfer
- Final Delivery

Email: Introduction

- Email jargons:
 - BTW (By The Way)
 - ROTFL (Rolling On The Floor Laughing)
 - IMHO (In My Humble Option)
- Some smileys. They will not be on the final exam :-).

Smiley	Meaning	Smiley	Meaning	Smiley	Meaning
: -)	I'm happy	=!:-)	Abe Lincoln	: +)	Big nose
: -(I'm sad/angry	=): -)	Uncle Sam	: -))	Double chin
: -	I'm apathetic	*<:-)	Santa Claus	: -{)	Mustache
; -)	I'm winking	<:- (Dunce	#:-)	Matted hair
: -(O)	I'm yelling	(-:	Australian	8-)	Wears glasses
: -(*)	I'm vomiting	: -)X	Man with bowtie	C:-)	Large brain

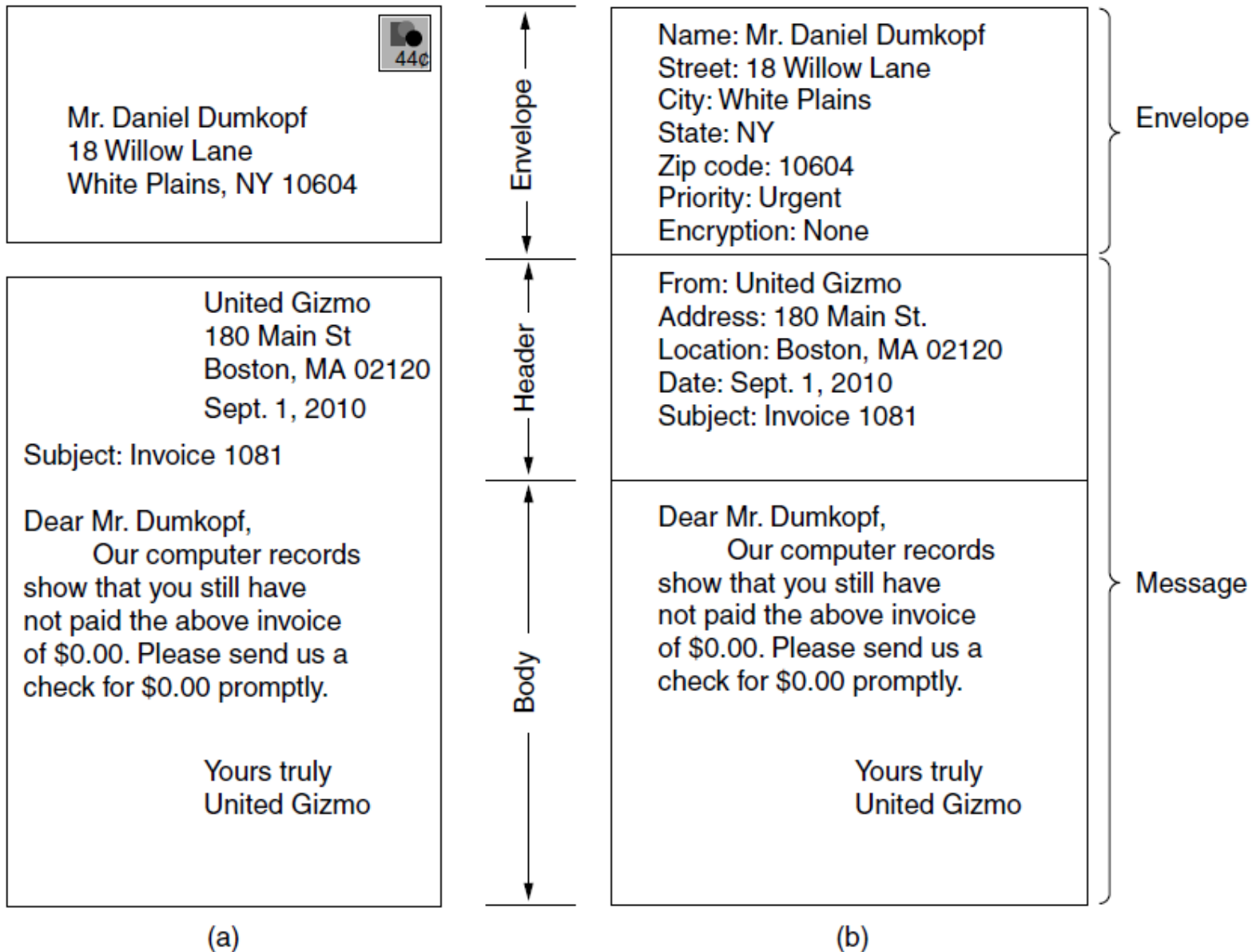
Email: Architecture and Services



Email: Architecture and Services

- **Basic functions of email systems:**
 - Composition: creating messages and answers
 - Transfer: moving messages from the originator to the recipient.
 - Reporting: telling the originator what happened to the message
 - Displaying: displaying emails.
 - Disposition: What the recipient does with the message after receiving it.
- **Additional functions of email systems**
 - Forwarding email, Mailing lists
 - Carbon copies, blind carbon copies
 - High-priority email, Secret email
 - Mailboxes
 - ...

Email: Architecture and Services



Email: User agent

Typical elements of the user agent interface

Message folders

Mail Folders
All items
Inbox
Networks
Travel
Junk Mail

Message summary

From		Subject	Received
trudy	✉	Not all Trudys are nasty	Today
Andy	📎	Material on RFID privacy	Today
djw	!	Have you seen this?	Mar 4
Amy N. Wong		Request for information	Mar 3
guido		Re: Paper acceptance	Mar 3
lazowska		More on that	Mar 2
lazowska	📎	New report out	Mar 2
...	

Search 🔍

Mailbox search

A. Student	Graduate studies?	Mar 1
Dear Professor, I recently completed my undergraduate studies with distinction at an excellent university. I will be visiting your		
...	...	

Message

Email: Message Formats

- Two kinds of Message Formats
 - RFC 5322 – The Internet Message format
 - Envelop fields vs message fields
 - ASCII message format
 - MIME (Multipurpose Internet Mail Extensions) message format

Email: Message Formats: RFC 5322

RFC 5322 header fields related to message transport.

Header	Meaning
To:	E-mail address(es) of primary recipient(s)
Cc:	E-mail address(es) of secondary recipient(s)
Bcc:	E-mail address(es) for blind carbon copies
From:	Person or people who created the message
Sender:	E-mail address of the actual sender
Received:	Line added by each transfer agent along the route
Return-Path:	Can be used to identify a path back to the sender

Email: Message Formats: RFC 5322

Some fields used in the RFC 5322 message header.

Header	Meaning
Date:	The date and time the message was sent
Reply-To:	E-mail address to which replies should be sent
Message-Id:	Unique number for referencing this message later
In-Reply-To:	Message-Id of the message to which this is a reply
References:	Other relevant Message-Ids
Keywords:	User-chosen keywords
Subject:	Short summary of the message for the one-line display

Email: Message Formats: MIME

- MIME – Multipurpose Internet Mail Extensions
- Problems with international languages:
 - Languages with accents
(French, German).
 - Languages with non-Latin alphabets
(Hebrew, Russian).
 - Languages without alphabets
(Chinese, Japanese).
 - Messages not containing text at all
(audio or images).

Email: Message Formats: MIME

Message headers added by MIME.

Header	Meaning
MIME-Version:	Identifies the MIME version
Content-Description:	Human-readable string telling what is in the message
Content-Id:	Unique identifier
Content-Transfer-Encoding:	How the body is wrapped for transmission
Content-Type:	Type and format of the content

Email: Message Formats: MIME

MIME content types and example subtypes.

Type	Example subtypes	Description
text	plain, html, xml, css	Text in various formats
image	gif, jpeg, tiff	Pictures
audio	basic, mpeg, mp4	Sounds
video	mpeg, mp4, quicktime	Movies
model	vrml	3D model
application	octet-stream, pdf, javascript, zip	Data produced by applications
message	http, rfc822	Encapsulated message
multipart	mixed, alternative, parallel, digest	Combination of multiple types

A multipart message containing enriched and audio alternatives.

C: From: alice@cs.washington.edu
C: To: bob@ee.uwa.edu.au
C: MIME-Version: 1.0
C: Message-Id: <0704760941.AA00747@ee.uwa.edu.au>
C: Content-Type: multipart/alternative; boundary=qwertyuiopasdfghjklzxcvbnm
C: Subject: Earth orbits sun integral number of times
C:
C: This is the preamble. The user agent ignores it. Have a nice day.
C:
C: --qwertyuiopasdfghjklzxcvbnm
C: Content-Type: text/html
C:
C: <p>Happy birthday to you
C: Happy birthday to you
C: Happy birthday dear <bold> Bob </bold>
C: Happy birthday to you
C:
C: --qwertyuiopasdfghjklzxcvbnm
C: Content-Type: message/external-body;
C: access-type="anon-ftp";
C: site="bicycle.cs.washington.edu";
C: directory="pub";
C: name="birthday.snd"
C:
C: content-type: audio/basic
C: content-transfer-encoding: base64
C: --qwertyuiopasdfghjklzxcvbnm

Email: Message Transfer: SMTP

- **SMTP** = Simple Mail Transfer Protocol
- Over the years, however, two different uses of SMTP have been differentiated.
 - mail submission: user agents send messages into the mail system for delivery.
 - transfer messages between message transfer agents
- SMTP is a simple ASCII protocol. This is not a weakness but a feature

SMTP: from *alice@cs.washington.com* to *bob@cs.washington.com*.

```
S: 220 ee.uwa.edu.au SMTP service ready
C: HELO abcd.com
S: 250 cs.washington.edu says hello to ee.uwa.edu.au
C: MAIL FROM: <alice@cs.washington.edu>
S: 250 sender ok
C: RCPT TO: <bob@ee.uwa.edu.au>
S: 250 recipient ok
C: DATA
S: 354 Send mail; end with "." on a line by itself
C: From: alice@cs.washington.edu
C: To: bob@ee.uwa.edu.au
C: MIME-Version: 1.0
C: Message-Id: <0704760941.AA00747@ee.uwa.edu.au>
C: Content-Type: multipart/alternative; boundary=qwertyuiopasdfghjklzxcvbnm
C: Subject: Earth orbits sun integral number of times
C:
C: This is the preamble. The user agent ignores it. Have a nice day.
C:
```

SMTP: from *alice@cs.washington.com* to *bob@cs.washington.com*. (Cont.)

C: --qwertyuiopasdfghjklzxcvbnm

C: Content-Type: text/html

C:

C: <p>Happy birthday to you

C: Happy birthday to you

C: Happy birthday dear <bold> Bob </bold>

C: Happy birthday to you

C:

C: --qwertyuiopasdfghjklzxcvbnm

C: Content-Type: message/external-body;

C: access-type="anon-ftp";

C: site="bicycle.cs.washington.edu";

C: directory="pub";

C: name="birthday.snd"

C:

C: content-type: audio/basic

C: content-transfer-encoding: base64

C: --qwertyuiopasdfghjklzxcvbnm

C: .

S: 250 message accepted

C: QUIT

S: 221 ee.uwa.edu.au closing connection

Email: Message Transfer: ESMTP

Some SMTP extensions

Keyword	Description
AUTH	Client authentication
BINARYMIME	Server accepts binary messages
CHUNKING	Server accepts large messages in chunks
SIZE	Check message size before trying to send
STARTTLS	Switch to secure transport (TLS; see Chap. 8)
UTF8SMTP	Internationalized addresses

Email: Final Delivery

- POP3 (Post Office Protocol v3)
- IMAP 4 (Internet Message Access Protocol v4)
- Webmail
 - Google Gmail
 - Microsoft Hotmail
 - Tencent Foxmail
 -

Email: POP3 vs. IMAP

Feature	POP3	IMAP
Where is protocol defined?	RFC 1939	RFC 2060
Which TCP port is used?	110	143
Where is e-mail stored?	User's PC	Server
Where is e-mail read?	Off-line	On-line
Connect time required?	Little	Much
Use of server resources?	Minimal	Extensive
Multiple mailboxes?	No	Yes
Who backs up mailboxes?	User	ISP
Good for mobile users?	No	Yes
User control over downloading?	Little	Great
Partial message downloads?	No	Yes
Are disk quotas a problem?	No	Could be in time
Simple to implement?	Yes	No
Widespread support?	Yes	Growing

Email: Final Delivery: IMAP

- IMAP (version 4) commands

Command	Description
CAPABILITY	List server capabilities
STARTTLS	Start secure transport (TLS; see Chap. 8)
LOGIN	Log on to server
AUTHENTICATE	Log on with other method
SELECT	Select a folder
EXAMINE	Select a read-only folder
CREATE	Create a folder
DELETE	Delete a folder
RENAME	Rename a folder
SUBSCRIBE	Add folder to active set
UNSUBSCRIBE	Remove folder from active set

Email: Final Delivery: IMAP

- IMAP (version 4) commands (Cont.)

LIST	List the available folders
LSUB	List the active folders
STATUS	Get the status of a folder
APPEND	Add a message to a folder
CHECK	Get a checkpoint of a folder
FETCH	Get messages from a folder
SEARCH	Find messages in a folder
STORE	Alter message flags
COPY	Make a copy of a message in a folder
EXPUNGE	Remove messages flagged for deletion
UID	Issue commands using unique identifiers
NOOP	Do nothing
CLOSE	Remove flagged messages and close folder
LOGOUT	Log out and close connection

Homework

1. Can a machine with a single DNS name have multiple IP addresses? How could this occur?
2. A binary file is 4560 bytes long. How long will it be if encoded using base64 encoding , with a CR+LF pair inserted after every 110 bytes sent and at the end?
3. When Web pages are sent out, they are prefixed by MIME headers. Why?
4. How do you make an image clickable in HTML? Give an example.