Chapter 2. Learning Objectives

- Understand host-based, client-based, and client-server architectures
- Understand how the Web works
- Understand how e-mail works
- Be aware of how FTP, Telnet, and Instant Messaging work
Chapter 2. Outline

• Introduction
• Application Architectures
  – Host-Based, Client-Based, and Client-Server
    Architectures, Choosing Architectures
• World Wide Web
  – How the Web Works, Inside an HTTP Request &
    HTTP Response
• Electronic Mail
  – How E-Mail Works, Inside an SMTP Packet
  – Listserv, Attachments in MIME
• Other Applications
  – Ftp, Telnet, Instant Messaging, Videoconferencing

Application Architectures

• Application programs can be thought of as being
  made up of four functions:
  – Data storage
  – Data access logic
  – Application logic
  – Presentation logic
• Application architectures are determined by how these
  functions are split between clients and servers.
• Three basic application architectures are in use today:
  – Host-based Architectures
  – Client-based architectures
  – Client-server architectures
Host-based Architectures

• The first application architecture to be developed was the host-based.
• The “server” in host-based architectures, usually a mainframe, performs all four application program functions.
• The client, usually a terminal or a PC running a terminal emulation program captures keystrokes which are sent to the mainframe and displays information according to the server’s instructions (Figure 2-1).

Figure 2-1 Host-Based Architectures

Host-based Architectures

• There are two main problems with host-based architectures:
  – Since all processing is by the host, the host becomes a bottleneck which can severely limit network performance.
  – Upgrades are typically expensive and “lumpy”, meaning available upgrades require big jumps in processing and memory. Network demand grows more incrementally, so this often means a poor fit (too much or too little) between host performance and network demand.
Client-Based Architectures

- Client-based architectures became important in the late 1980s with the widespread use of PCs, local area networks, and low-cost PC-based application programs such as spreadsheets and word processors.
- In client-based architectures, application programs on the clients are responsible for the data access, application, and presentation logic.
- The server is responsible for data storage only (Figure 2-2).

Client-Based Architectures

- The main problem with client-based architectures is the need for all of the data to travel back and forth between server and client.
- For example, if the client program is making a database query, the entire database must travel to the client before the query can be processed.
- The result can be poor network performance due to the bottleneck created by constantly sending large files back and forth over the network.
Client-Server Architectures

- In client-server architectures the application program functions are divided up between clients and servers.
- The client takes care of the presentation logic.
- The server handles data storage and data access logic.
- Application logic may reside on the client, server or be split up between the two.
- Most networks today use a client-server architecture (see Figure 2-3).

Figure 2-3 Client-Server Architectures

Client-Server Architectures

- Client-server architectures are more efficient since they distribute processing between client and server.
- Another strength is that they allow hardware and software from different servers to be used together.
- This is also a weakness, since it is sometimes difficult to get software from different vendors to work together smoothly.
- For this reason, a third category of software, called *middleware* was developed.
Middleware

- Middleware provides an interface for allowing client application programs to interact with server application programs.

- Middleware’s two main functions are:
  - Providing a standard way of translating between software from different vendors.
  - Managing message transfer between client and server, thus insulating network changes (such as adding a new server) from the clients.

- Two of the most important middleware standards are: Distributed Computing Environment (DCE) and Common Object Request Broker Architecture (CORBA).

Two-tier, Three-tier and N-tier Architectures

- Figure 2-3 gives an example of a 2-tier client-server architecture.

- In a three-tier architecture the application program logic is split up between three computers. For example (see Figure 2-4),
  - the client handles the presentation logic.
  - an application server handles the application logic.
  - a database server handles the data storage and data access logic.

- In an N-tier architecture more than three sets of computers are used (see Figure 2-5).
Two-tier, Three-tier and N-tier Architectures

- The primary advantage of N-tier architectures is that they make load balancing possible. Since the application program logic is distributed between several servers, processing can then be more evenly distributed among those servers.
- N-tiered architectures are also more easily scalable, since only servers experiencing high demand, such as the application server, need be upgraded.
- The primary disadvantage is that more distributed processing means a more heavily loaded network.
- It is also more difficult to program and test an N-tier architecture due to its increased complexity.

Thin versus Fat Clients

- Thin and Fat (or thick) clients describe how much of the application logic resides on the client.
  - Little or no application logic resides on a thin client.
  - All or most of the application logic resides on a fat client.
- Thin clients are becoming popular because they are easier to manage, since only the server application logic generally needs to be updated.
- The best example of a thin client is World Wide Web architecture which uses a two-tier, thin client architecture (Figure 2-6).
Best Practice Architecture

- Often, the application architecture is given, so no decision has to be made. When it is not, three important criteria for choosing an architecture are:
  - **Cost of Infrastructure**: Mainframes are expensive, which is why client-based and client-server architectures have become popular.
  - **Cost of Development**: Because software is expensive to develop, client-based and client-server architectures that use off-the-shelf software tend to be much cheaper than software solutions that require in-house development.
  - **Scalability** refers to the ability to increase (or decrease) in computing capacity as network demand changes. Client-server architectures are the most scalable since both clients and servers can be added to meet network needs.

### Figure 2-6 Web Architecture

<table>
<thead>
<tr>
<th></th>
<th>Host-Based</th>
<th>Client-Based</th>
<th>Client-Server</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cost of Infrastructure</td>
<td>High</td>
<td>Medium</td>
<td>Low</td>
</tr>
<tr>
<td>Cost of Development</td>
<td>Low</td>
<td>Medium</td>
<td>High</td>
</tr>
<tr>
<td>Scalability</td>
<td>Low</td>
<td>Medium</td>
<td>High</td>
</tr>
</tbody>
</table>

### Figure 2-7 Choosing an Architecture
Origin of the World Wide Web

- Invented in 1989 by Tim Berners-Lee at the Centre Européan pour Recherche Nucléaire (CERN) in Geneva.
- Two central WWW ideas are hyperlinks and Uniform Resource Locators (URLs).
- Mosaic, the first web reader (browser) to gain widespread use, was written in 1993 at the National Center for Supercomputing Applications (NCSA) by Marc Andressen, who later founded Netscape.

How the Web Works (Figure 2-8)

- Based on a Client-Server architecture
- The main Web communications protocol is Hypertext Transfer Protocol.
- Web clients (browsers) make HTTP requests to Web servers. Web servers answer with HTTP responses.
- Clicking on a hyperlink or typing a URL into a browser starts an HTTP request-response cycle.
- An HTTP request-response cycle can have multiple steps since many web pages often contain embedded files, such as graphics, each requiring a separate response.
HTTP Requests have 3 parts:

- **Request line** (required): starts with a command and URL and ends with the http version number.
- **Request header** (optional): contains information on the browser, date, and the referring page from which the request originated. If the user types in the URL by themselves, then the referring page is blank.
- **Request body** (optional): contains information sent to the server, such as from a form.
HTTP Responses also have 3 parts:

- **Response status** (optional): http version number, status code (e.g., 200 = ok, 404 = page not found) and “reason” phrase (description of status code).
- **Response header** (optional): contains information on the server, date, exact URL of the retrieved page and format used for the body of the response (e.g., html).
- **Response body** (required): contains the requested Web page.

![HTTP Response](image)

**Hypertext Markup Language (HTML)**

- Web pages are formatted using another protocol, usually HTML, also developed at CERN.
- HTML documents include information on how to format the file imbedded in the document called tags.
- HTML was initially developed for text files, not multimedia files. This is why extensions to HTML have been needed.
- And also why other markup language formats have been created, such as Extensible Markup Language (XML).
Electronic Mail

• Still the most heavily used Internet application because:
  – It’s much, much faster than regular mail.
  – It’s extremely inexpensive.
  – It can substitute for other forms of communication, such as telephone calls.
  – E-mail users don’t have to play “telephone tag”, since the respondent can answer at his/her convenience, instead of time of call.

E-mail standards

• Simple Mail Transfer Protocol (SMTP): part of the TCP/IP protocol suite.
• SMTP is the main e-mail standard for mail transfer agents in use today, but was written to only handle text files.
• SMTP is usually implemented using a two-tier client-server architecture.
• Two other competing standards for e-mail are Common Messaging Calls (CMC) and X.400.
Two-Tier E-mail Architecture

- SMTP uses two types of programs:
  - **User agents**, also called e-mail clients, are programs running on client computers that send e-mail to e-mail servers and download e-mail from mailboxes on those servers.
    - Post Office Protocol (POP) & Internet Mail Access Protocol (IMAP) are the main protocols for user agent to mail server communications (e.g., Eudora, Outlook).
  - **Mail transfer agents**, also called mail server software, are used by e-mail servers. They send e-mail between e-mail servers and maintain individual mailboxes.

Sending An E-mail Message (Figure 2-11)

- When completed, an e-mail message is sent by the sender’s e-mail client as an SMTP packet to the local mail server.
- The mail server’s message transfer agent next reads the packet’s destination address and sends it over the Internet to the receiver’s mail server.
- The destination mail transfer agent then stores the message in the receiver’s mail box.
- When the receiver next accesses e-mail, his or her user agent contacts the local mail server which then downloads the message to the receiver’s client computer.

Figure 2-11 How SMTP e-mail works
Three-Tier Client-Server Architecture (Figure 2-12)

- Reading e-mail using a Web-based interface, such as Hotmail, uses a three-tier architecture. The three tiers are:
  - The **client's web browser** that sends HTTP requests to the Web server.
  - The **Web server**: 1) sends HTTP responses to the Web client and 2) translates the client’s HTTP requests into SMTP packets which are then sent to the Mail server.
  - The **Mail server** performs the same functions as the mail server in the two-tier example.

Simple Mail Transfer Protocol

- SMTP E-mail messages have the following structure:
  - **Header**: lists source and destination addresses, date, subject, and other information about the e-mail message.
  - **Body**: the message itself.
  - **Attachments**: additional files that have been included along with the e-mail message.
Attachments in MIME

- Since SMTP was developed for transferring text files, special techniques needed to be developed to send graphical information.
- Graphics capable mail transfer agent protocols include MIME (Multipurpose Internet Mail Extension), uuencode and binhex.
- Included as part of an e-mail client, MIME translates graphical information into text allowing the graphic to be sent as part of an SMTP message.
- The receiver’s e-mail client then translates the MIME attachment from text back into graphical format.

Other Applications
File Transfer Protocol (FTP) and Telnet

- FTP enables sending and receiving files over the Internet.
- Telnet allows remote logins over the Internet.
- Both were much more commonly used during the pre-WWW days of the Internet.
- FTP is still commonly used today for uploading web pages.

IM: Instant Messaging (Figure 2-14)

- IM is a client-server program that allows real-time messages to be exchanged. Some types also allow voice and video packets to be sent. Examples include AOL and ICQ.
- IM works as follows:
  - Your IM client first sends a request to the IM server telling it that you are online. If a friend connects, the IM server sends a packet to your IM client and vice versa.
  - When you type some text, your IM client sends the text in a packet to the IM server which relays it to your friend.
  - If a chat session has more than two clients, multiple packets are sent by the IM server. IM servers can also relay information to other IM servers.
Videoconferencing

- Allows people to “meet” and interact at the same time but in different locations, typically saving money and time by not having people move around to attend meetings.
- Conference calling does the same thing.
- Usually involves 2 meeting rooms only.
- Desktop videoconferencing is a related, low cost application linking small video cameras and microphones together over the Internet.

End of Chapter 2