Outline

- Announcements
- Review Layering
- Student Presentation
- Client-Server Computing
- Student Presentation
- SUN Case
Announcements

- News Folio 2 DUE today
  - Include a print/copy of the article

- Homework 3 out and is due 2/17/2011

- Next time read:
  - Chapter 6 Messerchmitt
Announcements

Next week’s presentations:

2/15
- Natalie Marie Cross Bus Proj: Coca-Cola
- Yue Siao Bus Proj: Starbucks

2/17
- Morgan Marie Hunt Bus Proj: Netflix
- Nicholas John Raimondi Bus Proj: Home Depot
Sources for business paper

- You should cite at least 5 non-web sources
  - Ideally a lot more!
  - A list of sources is included in 1st lecture’s slides

- If an article is available on-line and also in print, you should cite the print version!!!!

- Only cite a website if you are sure there is no print version of the material available.
  - Your citation of the print version may also indicate a website where the same article is available.

- Wikipedia is a nice source for gaining some background knowledge on your research or find links to other sources BUT is not always 100% accurate/true
Citing Sources

- More than thirty words verbatim must be cited.

- Any facts or figures that are not your own must be cited in the body of the text!!!
  - Ebay’s revenues in US Revenues in 2002 were $1.39 billion [1].

- Any Tables of figures must be cited!!
  - You can create tables/charts based on your findings
“Companies that have deployed Internet technology have been confused by distorted market signals, often of their own creation.”¹


References:

**Review: Simplified infrastructure layering**

<table>
<thead>
<tr>
<th>Communications</th>
<th>Network equipment</th>
<th>Network software</th>
<th>Distributed object management</th>
<th>Database management</th>
<th>File system</th>
<th>Storage peripherals</th>
<th>Storage</th>
<th>Middleware</th>
<th>Operating system</th>
<th>Equipment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Emergence

Subsystems are more specialized and simpler functionality

Higher-level system functionality arises from the interaction of subsystems

**Emergence** includes capabilities that arise purely from that interaction (desired or not)

- e.g. airplane flies, but subsystems can’t
Layering builds capability incrementally by adding to what exists.
Announcements

- **Database Tutorials**
  - MSAccess
  - Will help you on your DB Assignment that will be out soon

- **3 labs**
  - You should register and attend ONLY ONE
  - To register visit class forum
  - Sign up with your name under the day/time that you prefer
  - First come-first served (if class is full then you will have to choose another day/time)
Announcements

- Database Tutorials

- Dates:
  - 5/14 (Wed), 1 - 3 PM, Baskin lab 109
  - 5/15 (Thurs), 2 - 4 PM, Baskin lab 109
  - 5/16 (Fri), 2 - 4 PM, Baskin lab 109

- Each student should attend ONLY ONE
Student Talks

- Project presentation (Ticketmaster)
  - Paul Krumholz
Client-Server Computing

Two host Architecture
Three Tier Client Server Architecture
N-Tier Client Server Architecture (Sun)
Two Host Architectures

- **Client-Server**
  - Attractive for information access and organizational applications

- **Peer-to-Peer (P2P)**
  - Appropriate for direct/immediate applications where no centralized application logic is needed
Two-tier client/server

desktop computer

Local-area network

Server/ Mainframe

file service
Peer to Peer (P2P)

Slide adapted from slides for Understanding Networked Applications by David G Messerschmitt. Copyright 2000. See copyright notice.
Peer to Peer (P2P)

- Each peer provides the same functionality
  - A P2P Network relies on
    - computing power
    - bandwidth of each peer
  - **Ad-hoc** connections: Each peer joins/leaves the network at any time

- **What is P2P good for?**
  - File sharing
  - Video Conferencing
  - Internet telephony
  - Etc.
3-tier Client/Server Architecture

- Many organizational applications incorporate mission-critical databases
- Need to separate data from the applications that access them
  - Centralized storage
  - Security
  - Fault tolerance
  - Data shared across many applications
Client

Host architecture

Web browser

HTTP

Web server

Application logic

Databases and DBMS

Application partition

Slide adapted from slides for *Understanding Networked Applications* by David G. Messerschmitt. Copyright 2000. See copyright notice.
Customer distribution centers

Books4u.com

Financial institution

Book distribution centers

Consumer

Enterprise

Inter-enterprise

Slide adapted from slides for *Understanding Networked Applications* By David G Messerschmitt. Copyright 2000. See copyright notice.
Inter-organizational computing

Global internet
E-commerce
Scalability

- Need to serve multiple customers at the same time
- Customer base can grow large

- Many identical application servers
  - Each user session carried by a single server
  - Each server can handle a limited number of clients

- More than one databases
  - Storing different kinds of data (e.g. customers, orders, products, etc.)
## C/S vs. P2P

<table>
<thead>
<tr>
<th>Type of computers:</th>
<th>Powerful computers used as servers. Different machines from clients.</th>
<th>Each peer is both a client and a server.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type of network:</td>
<td>Predefined number/topology of servers. Clients can connect/leave any time.</td>
<td>Ad-hoc connections.</td>
</tr>
<tr>
<td>Communication:</td>
<td>To and from the server.</td>
<td>Among peers.</td>
</tr>
<tr>
<td>Network relies on:</td>
<td>Computing power/availability of a limited number of servers.</td>
<td>Computing power/Bandwidth of each peer.</td>
</tr>
</tbody>
</table>

- Most P2P applications also have a C/S part
  - E.g Napster, IRC
  - P2P File sharing/Transfer
  - C/S searching
Sun Case

- Student presentation
  - Sarah James
Sun Case
(N-tier C/S)

- What does Sun make?
  - Workstations
  - Servers
  - Software
How Successful had Sun been up to 1998?

- Founded in 1982
- Open Standards Workstation
  - Unix Operating System (Solaris)
- 1988 – Revenues $1 billion
- 1993 – Market value $3.6 billion
  - Known for its workstations (addressing engineers)
  - Achieved a 75% year-over-year increase in total server shipments
- Designed its own hardware/software:
  - SPARC microprocessor, Solaris OS, Networking capabilities (TCP/IP)
- Scalable, Reliable network computing for large corporations
  - ERP systems
  - Intranets (SCM, email, file directories etc)
Java

- 1995 - Sun introduces Java
- Programming Language
  - Portable between computers with different hardware/operating system platforms
  - JVM: Java Virtual Machine
  - Easy to write programs in
  - Easier re-use: “Write once, run everywhere”

- But, programs were slow
- Constant updates on JDK library
  - Programmers had to update to current versions
- Some felt that language was not yet mature
Java Applets

- Key feature of Java

- **Applets**: chunks of Java code
  - Usually embedded in other applications, e.g. Web Pages
  - Initially enabled animations on web pages
  - Later used to facilitate e-commerce applications, in cellular phones, etc.

- **Applets are downloaded through the browser**
  - Only what and when was needed
  - No need to keep a copy on client!

- **Servlets**
  - One can think of them as server-side applets
In the meantime, Microsoft...

- **Dominated Desktop software (mid 90’s)**
  - Users familiar with Windows, Office, etc.

- **Windows NT servers**
  - Fine for small intranets, “not industrial strength”
What problems did the micro era produce?

- Desktops are expensive to maintain
  - TCO for windows PC $9900!

- Every PC had a lot of software that had to be maintained
  - Office, Windows, etc...

- Small differences, like the order in which software is installed, could make different PCs behave differently!

- Keep all PCs in organization updated with current/same software releases
  - Compatibility issues
In the Networking Era...

- These “bloated” PCs are networked and termed **fat clients**.

- But networking of PCs offered the possibility of
  - putting most of the functionality into servers
  - getting rid of much of the software on the client

- These clients would be called **thin clients**.
  - Sun, Oracle, and others saw it as the future.
Hardware for thin clients

- A **Network Computer** (NC) - a computer with minimal hardware that depends on a network connection to a server to function
  - Trademark of ORACLE, used by ORACLE, SUN, Acron
  - Be careful not to confuse it with the phrase “networked computer!”
  - Example: Sun’s JavaStation (1996-2000)

- It is the hardware one would use to implement a *thin-client* computing model:
  - Store desktop on the network
  - Send desktop via browser to the client

- Much less unique “instances” of working environment
In the meantime...

- **NetPC** was a PC introduced by Microsoft and Intel in 1996
  - Same software as a normal PC
  - Did not allow users to install their own software
  - NetPC died out
  - Features of it, and Microsoft's Zero Administration Kit, live on in today's version of Windows.
Microsoft Vision

- Keep “fat-client” model
- Add some features to Windows to reduce administration costs
Microsoft vs. Java

- Announced that will “embrace and extend” promising technologies and emerging open standards
- **1996: Sun licenses Java to Microsoft** to integrate into Explorer
  - Microsoft downplays Java’s importance
  - Did not deliver compatible implementation of Java in its products
    - Customers were frustrated since the Java-components would not work
- **Includes Java in its programming platform**
  - BUT incorporates proprietary components making it impossible to run on different OS
Sun’s Vision

- “Thin Client” model
- Application Servers with Applications written in Java
  - Applets/Servlets
- NCs could retrieve applications from application server as needed
- Applications compatible with any NC hardware and OS
- Applications could be fixed, added, updated at the server level, rather than maintaining each PC
SUN 3 - Tier

Exhibit 1  Three-tier Architecture

Tier Three

Database

Tier Two

Applets
App Server

Tier One

Client Applets  Client Hardware

These had to be managed locally. If code needed to be updated, each app server had to be shut down, updated and rebooted.

Client Applets  Client Hardware

JDBC: Stands for Java Database Connectivity. It is a programming interface that lets Java applications access a database via the SQL language. RMI: Stands for Remote Method Invocation. It is the method by which a remote Java object from one location can be invoked from other Java virtual machines. HTTP: Stands for HyperText Transport Protocol. It is the communications protocol used to connect to servers on the World Wide Web.
Sun N-tier
Sun N-Tier

**Step 1:** The user logs into his client and calls down an application. This message is sent to the Application Server.

**Step 2:** An initial applet is sent to the client. At the same time a servlet is sent to the Webtop Server.

**Step 3:** The applet talks back and forth with the Webtop Server via the LAN.

**Step 4:** As new data is received (i.e., a new customer’s name) the App Server communicates with the database to update that information.

Remote: The database and App Server communicate with the Webtop Server via a WAN.

Local: The Webtop Server and client communicate via a LAN.

**Exhibit 3** How the N-tier Architecture Works
Sun’s Performance

Net Income  355.8   476.4   762.4   762.9   1,031.3   1,854.0   927.0   (587.0)   (3,429.0)   (388.0)   (106.0)
Today

- 3-tier model common.
- Sun’s version of 4-tier model not-common.
- N-tier model where Webserver and Application Server on separate equipment also common.
- Sun’s hardware business not strong.
  - Linux on cheap PCs most common servers
  - Microsoft desktops replacing Sun workstations
Today

- **Oracle bought Sun**
- **Java**
  - Common in Server implementations
    - Example: Java Servlet implementing application logic in a banking application.
  - Often used to push simple applets onto client
  - Not common
    - For “big” desktop applications
    - Other languages, e.g. C/C++ still faster
  - **Microsoft is still in business...**
Modularity and Layering
Application Architecture Design

- The most important step
  - Hardest to change
  - Influences everything that follows

- Conceptualization
  - What is it you are trying to do?

- Example Concept:
  - Small HHC for flight attendants.
  - HHC tells flight attendants which passengers are higher priority.
    - Who paid the highest fares
    - Who has been a more valuable customer in past
  - Flight attendant discriminates based on this
    - Free drinks, meals, and pillows to valuable customers
Example Concept:
Architecture

- What is the complexity of such a problem?
- How do you begin to architect a solution for a problem like this?
- Break it into modules!
- What is a “good” architecture?
When a module is composed of sub-modules, the architecture is **hierarchical**.
We are using a *layered architecture* as well.
Allows reuse of previously built infrastructure.
Properties of Modularity

- Functionality
- Hierarchy
- Separation of concerns
- Interoperability
- Reusability
Granularity tradeoff

- How big should we make the modules
  - Many simple small ones... (*fine granularity*)
  - Or a few complicated big ones... (*coarse granularity*)

- Which is better?
  - *Coarse Granularity*: Fewer things to keep track of
  - *Fine Granularity*: Modules are simpler and easier to understand

- *Hierarchy* of modules: Allows the system to be viewed at different granularity levels.