Class announcements

- **Tuesday:**
  - Messerschmitt Ch 18 (493-512)

- **For Thursday 3/4:**
  - Database assignment due
Student Presentation
Stovepipe vs. Integrated Infrastructure

**stovepipe architecture**

---or---

**Turnkey Solution**

- Single supplier provides all encompassing solution
- (complete with infrastructure)

**Integrated Infrastructure**

- Separate infrastructure that can support many applications

---

Slide adapted from slides for *Understanding Networked Applications*
By David G Messerschmitt. Copyright 2000. See copyright notice.
From stovepipe to layering

Data
Voice
Video

Application-dependent infrastructure

Many applications
Integrated Infrastructure
(Maybe broken into Additional layers.)

Application-independent
Stovepipe vs. Integrated Infrastructure

- What are some examples of each?

- What are the advantages of each approach?
Vertical Integration vs. Diversification

- A company is *vertically integrated* when it makes rather than buys the subsystems in its products.

- A *diversified* company produces products across different industry segments.
Vertical Integration vs. Diversification

- Why do customers favor less vertical integration?
  - Prefer competition amongst component suppliers
  - Mix and match components
  - Reduced lock in

- Disadvantages??
  - Customer needs to integrate components from different suppliers.

Slide adapted from slides for *Understanding Networked Applications* by David G Messerschmitt. Copyright 2000. See copyright notice.
Vertical Integration vs. Diversification

Why do customers favor diversification?

- Reduce coordination costs by having to deal with fewer suppliers.
General Trend

- Less Vertical Integration
- More Diversification
- Of course there are exceptions...
Today's supplier structure

- Applications
- Frameworks and components
- Middleware
- Infrastructure (network, OS) software
- Equipment (network, computers)
- Semiconductors, components

Slide adapted from slides for Understanding Networked Applications
By David G Messerschmitt. Copyright 2000. See copyright notice
Standardization
Purpose of a standard?

- Allow products or services from different suppliers or providers to be interoperable
Scope of a standard

Included:
- interfaces (physical, electrical, information)
- architecture (reference model)
- formats and protocols (FAP)
- compliance tests (or process)

Excluded:
- implementation
- (possibly) extensions
Reference model

Decide decomposition of system
- where interfaces fall

Defines the boundaries of competition and ultimately industrial organization
- competition on the same side of an interface
- complementary suppliers on different sides
- hierarchical decomposition at the option of suppliers
- (possibly) optional extensions at option of suppliers

Slide adapted from slides for Understanding Networked Applications
By David G Messerschmitt. Copyright 2000. See copyright notice
Some issues

Once a standard is set

- becomes possible source of industry lock-in; overcoming that standard requires a major (~10x?) advance
- may lock out some innovation

In recognition, some standards evolve

- IETF, CCITT (modems), MPEG
- backward compatibility
Types of standards

*de jure*
- Sanctioned and actively promoted by some organization with jurisdiction, or by government

*de facto*
- Dominant solution arising out of the market
- Voluntary industry standards body

Industry consortium
Common or best practice

Examples?
Examples

\textit{de jure}
- GSM, ISDN Telephone interface

\textit{de facto}
- Microsoft Windows API (Application Programming Interface)
- Intel Pentium instruction set,

\textit{Voluntary industry standards body}
- IEEE (Institute of Electrical and Electronic Engineers)
- IETF (Internet Engineering Task Force)

\textit{Industry consortium}
- W3C (World Wide Web Consortium)
- SET (Secure Electronic Transactions)

\textit{Best practice}
- Windowed GUI

Slide adapted from slides for \textit{Understanding Networked Applications}
By David G Messerschmitt. Copyright 2000. See copyright notice
The changing process

- As technology and industry move more quickly, the global consensus standards activity has proven too unwieldy
  - e.g. ISO

- “New age” standards activities are more informal, less consensus driven, a little less political, more strategic, smaller groups
  - e.g. OMG, IETF, ATM Forum, WAP

Programmable/extensible approaches for flexibility
- e.g. XML, Java
Old giving way to the new

The Standards Making Universe

Traditional Model Telco Bodies

International Coordinating Organizations

Traditional Model Information Systems Bodies

Traditional Radio Bodies

New Model Telco Bodies

New Model Information Systems Bodies

Slide adapted from slides for *Understanding Networked Applications* By David G Messerschmitt. Copyright 2000. See copyright notice
Reasons for change

- From government sanction/ownership to market forces
  - Increasing fragmentation
  - Importance of time to market

Greater complexity

- Less physical/performance constraint for either hardware or software
Lock-in

(Particularly open) standards reduce consumer lock-in

- Consumers can mix and match complementary products

Increase supplier lock-in

- Innovation limited by backward compatibility
  - e.g. IP/TCP, x86, Hayes command set

Slide adapted from slides for Understanding Networked Applications
By David G Messerschmitt. Copyright 2000. See copyright notice
Aside: Network Effects

- The value of owning some products goes up if lots of other people have it too.
  - Examples?

- This phenomenon is called “network effects”

- How do standards influence network effects?
Network effects

Standards can harness network effects to the industry advantage

- Revenue = (market size) x (market share)

Increases value to customer

Increases competition

- Only within confines of the standard
- But forces customer integration or services of a system integrator
Why standards?

*de jure* are customer driven to reduce confusion and cost
*de facto* standards are sometimes the result of positive feedback in network effects

Customers and suppliers like them because they
- increase value
- reduce lock in

Governments like them because they
- promote competition in some circumstances
- *May believe* they can be used to national advantage

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

Consensus
- ISO

Collaborative design
- MPEG

Competitive “bake off”
- IETF

Coordination of vendors
- OMG
Open vs. Proprietary Standards

- Open standard – a standard that is well documented, unencumbered by intellectual property rights and restrictions, and available to any vendor.

- What are the advantages?

- What are the disadvantages?
Why companies participate

Pool expertise in collaborative design
  - e.g. MPEG

Have influence on the standard

Get technology into the standard
  - Proprietary, with expectation of royalties
  - Non-proprietary

Reduced time to market
Standards applied to Business Processes?

- Can you standardize business processes?

- Yes!
  - ISO 9000
    - A set of standardized business processes for Quality Management.
    - Supports TQM (Total Quality Management)
  - RosettaNet
    - A set of standardized business processes, and accompanying standardized data interfaces/formats for conducting e-business.
Databases

by

David G. Messerschmitt
Databases

Treat data as a separate asset
- May be shared by multiple applications

Provide protection and integrity features appropriate to mission-critical data
- Access control
- Integrity constraints
- Persistence
- etc.
Two capabilities

**Aggregation:** accessing multiple databases

**Sharing:** two or more applications accessing the same databases
Relational table

<table>
<thead>
<tr>
<th>Table</th>
<th>Employee</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Address</td>
</tr>
<tr>
<td>Record</td>
<td>Field</td>
</tr>
</tbody>
</table>

The image illustrates a relational table with columns for Name, Address, and Dept, and rows for Employee records.
SQL interface

- SQL (Structured Query Language)
- Presents single abstract interface to the application logic
  - For manipulating, and extracting data from database
- Standardized, not vendor specific

- Encapsulates various internal details
  - Data partitioning and replication
  - Host mapping
  - File representation
  - etc.
Database operations

Each operation results in a new table
Multiple tables
## Database Operations

### Passengers

<table>
<thead>
<tr>
<th>Name</th>
<th>Dept ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alice</td>
<td>1</td>
</tr>
<tr>
<td>Bob</td>
<td>1</td>
</tr>
<tr>
<td>Chris</td>
<td>2</td>
</tr>
</tbody>
</table>

### Departments

<table>
<thead>
<tr>
<th>Dept Name</th>
<th>Dept ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>Engineering</td>
<td>1</td>
</tr>
<tr>
<td>Sales</td>
<td>2</td>
</tr>
</tbody>
</table>

### JOIN

<table>
<thead>
<tr>
<th>Name</th>
<th>Dept ID</th>
<th>Dept Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alice</td>
<td>1</td>
<td>Engineering</td>
</tr>
<tr>
<td>Bob</td>
<td>1</td>
<td>Engineering</td>
</tr>
<tr>
<td>Chris</td>
<td>2</td>
<td>Sales</td>
</tr>
</tbody>
</table>
### Fields, columns, attributes

<table>
<thead>
<tr>
<th>Year</th>
<th>City</th>
<th>Accommodation</th>
<th>Tourists</th>
</tr>
</thead>
<tbody>
<tr>
<td>2002</td>
<td>Oakley</td>
<td>Bed&amp;Breakfast</td>
<td>14</td>
</tr>
<tr>
<td>2002</td>
<td>Oakley</td>
<td>Resort</td>
<td>190</td>
</tr>
<tr>
<td>2002</td>
<td>Oakland</td>
<td>Bed&amp;Breakfast</td>
<td>340</td>
</tr>
<tr>
<td>2002</td>
<td>Oakland</td>
<td>Resort</td>
<td>230</td>
</tr>
<tr>
<td>2002</td>
<td>Berkeley</td>
<td>Camping</td>
<td>120000</td>
</tr>
<tr>
<td>2002</td>
<td>Berkeley</td>
<td>Bed&amp;Breakfast</td>
<td>3450</td>
</tr>
<tr>
<td>2002</td>
<td>Berkeley</td>
<td>Resort</td>
<td>390800</td>
</tr>
<tr>
<td>2002</td>
<td>Albany</td>
<td>Camping</td>
<td>8790</td>
</tr>
<tr>
<td>2002</td>
<td>Albany</td>
<td>Bed&amp;Breakfast</td>
<td>3240</td>
</tr>
<tr>
<td>2003</td>
<td>Oakley</td>
<td>Bed&amp;Breakfast</td>
<td>55</td>
</tr>
<tr>
<td>2003</td>
<td>Oakley</td>
<td>Resort</td>
<td>320</td>
</tr>
<tr>
<td>2003</td>
<td>Oakland</td>
<td>Bed&amp;Breakfast</td>
<td>280</td>
</tr>
<tr>
<td>2003</td>
<td>Oakland</td>
<td>Resort</td>
<td>210</td>
</tr>
<tr>
<td>2003</td>
<td>Berkeley</td>
<td>Camping</td>
<td>115800</td>
</tr>
<tr>
<td>2003</td>
<td>Berkeley</td>
<td>Bed&amp;Breakfast</td>
<td>4560</td>
</tr>
<tr>
<td>2003</td>
<td>Berkeley</td>
<td>Resort</td>
<td>419000</td>
</tr>
<tr>
<td>2003</td>
<td>Albany</td>
<td>Camping</td>
<td>7650</td>
</tr>
<tr>
<td>2003</td>
<td>Albany</td>
<td>Bed&amp;Breakfast</td>
<td>6750</td>
</tr>
</tbody>
</table>

- Entries are simple data types or compositions of those types
  - Integer, string, etc.
Object-relational database

- A column can store object instances of a given class rather than data of a given simple or compound data type
- Because of the table structure, SQL can be extended to this case
- Standard SQL queries can be extended to methods returning simple data types
- Many other good ideas
Benefits of ORDBMS

Extension: manage arbitrarily complex data types
Migration: preserve and extend existing databases
Preserve SQL interface
  - OR extensions in latest standard
All the benefits/experience of earlier databases
  - Access control, data integrity, persistence, etc.
Killer app: Behind Web/CGI
  - Images, video, audio, animation, applets, etc.
Markup languages
Definition

A *markup language* describes the structure of a document

- Based on tags
- Tags denote structural elements like sections, subsections, figures, etc

Internationally standardized, so application independent
Example: HTML

<html>
<h1>Super Widget</h1>
<h2>Widgets Incorporated</h2>
<em>123456789</em>
<br/>
<p>$300</p>
</html>

Super Widget

Widgets Incorporated

123456789

$300
Example: XML

Tags Emphasize what the things *mean* rather than how to *format* their Presentation.

```xml
<xml>
  <product>
    <model> Super Widget </model>
    <make> Widgets Incorporated </make>
    <sku> 123456789 </sku>
    <price> $300 </price>
  </product>
</xml>
```
XML in Ecommerce example

```xml
<xml>
  <product>
    <model> Super Widget </model>
    <make> Widgets Incorporated </make>
    <sku> 123456789 </sku>
    <price> $300 </price>
  </product>
</xml>
```

Product info
From each Supplier sent in XML

Supplier

<table>
<thead>
<tr>
<th>Stuff4U</th>
</tr>
</thead>
<tbody>
<tr>
<td>Super Widget</td>
</tr>
<tr>
<td>Amazing Gadget</td>
</tr>
</tbody>
</table>

Retailer

Consumer
XML in ecommerce example 2

Supplier

From each Supplier sent in XML

<Product info>

<product>
  <model> Super Widget </model>
  <make> Widgets Incorporated </make>
  <sku> 123456789 </sku>
  <price> $300 </price>
</product>

Super widget recognized and managed by SCM software.
Family lineage

SGML

- Standardized in mid 80s by ISO

HTML

- Introduced in Early 90s
- Emphasizes formatting and presentation of documents

XML

- Proposed in mid 90s
- Emphasizes structure of documents
- Purpose- and industry-specific extensions
Break!
mySQL Case
MySQL Quiz!!!

Take out a sheet of paper, and answer these questions:

1) What does MySQL make?

2) In 2003, MySQL formed an alliance with what major enterprise software company?

3) According to the case, what operating system was leading the open source software movement?
   a) Linux    b) Windows    c) MAC OS    d) DOS
mySQL student talk
mySQL

What does mySQL make?

How Successful is mySQL?
- Visibility: Fortune magazine, more mentions on www
- Reaction from giants
- Revenue growth 2001 700k, 2002 6.2m, 2003 10m
- Good performance reviews
- Recent SAP alliance
- But Market share tiny:
  - $10 million out of $10 billion market!

Why Success?
- Good Technology
- Large DBMS bloated with features most don’t need
- Innovative OSS model
mySQL

How does OSS work?

Two Types of License:
- **GPL**
  - Free
  - No Support
  - Any software that uses MySQL as a module must itself be made GPL
- **Commercial License**
  - Support
  - Could be distributed with non-open source software
  - Not Free:
    - MySQL: Classic $250, Pro $495 (for ~ 50 users)
    - Compare to:
      - MSFT  $3150  single proc for 50 users
      - IBM   $33000 single proc for 50 users
      - Oracle $40000 single proc for 50 users
Aside: DB’s in different software stacks

<table>
<thead>
<tr>
<th>General Software Stack</th>
<th>ERP Software Stack</th>
<th>Web Application Software Stack</th>
<th>Banking Software Stack</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Or Oracle, Axtapa, etc.</td>
<td>Apache Web Server</td>
<td></td>
</tr>
<tr>
<td>Middleware</td>
<td>Oracle</td>
<td>MySQL</td>
<td>Oracle</td>
</tr>
<tr>
<td></td>
<td>or MySQL, IBM, etc</td>
<td>or other DB</td>
<td>or other DB</td>
</tr>
<tr>
<td>Operating System</td>
<td>MS Windows</td>
<td>Linux</td>
<td>IBM z/OS</td>
</tr>
<tr>
<td></td>
<td>or other OS</td>
<td>or other OS</td>
<td>or other OS</td>
</tr>
</tbody>
</table>

- Which companies are competitors?
- Which are complimenters?
- Which are both!?
MySQL

- Which segments of market is mySQL strong in?
  - Large Companies or Small Companies?
  - Web applications or Critical Enterprise data?

- Why would a major enterprise want to pay so much more for an Oracle or IBM DB?
## My SQL: market

<table>
<thead>
<tr>
<th></th>
<th>Small 20%</th>
<th>Medium 30%</th>
<th>Large 50%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enterprise</td>
<td>Microsoft</td>
<td>Oracle</td>
<td>IBM</td>
</tr>
<tr>
<td>wide data</td>
<td>90%</td>
<td>Reliability</td>
<td>Scalability</td>
</tr>
<tr>
<td>90%</td>
<td></td>
<td>Support</td>
<td>Longevity</td>
</tr>
<tr>
<td>Web Sites</td>
<td>My SQL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10%</td>
<td>Cost</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

How should mySQL grow in order to meet it’s stated goal of getting to $100 million in revenue?

Figure Adapted from “Teaching Note for MySQL Open Source Database,” 6/1/04, Stanford GSB.
**My SQL: Growth Strategy**

- Lack of Brand identity in this segment
- MySQL lacks the organization to offer support
- Large enterprises have high switching costs

Figure Adapted from “Teaching Note for MySQL Open Source Database,” 6/1/04, Stanford GSB.
### My SQL: Growth Strategy

<table>
<thead>
<tr>
<th>Enterprise wide data 90%</th>
<th>Small 20%</th>
<th>Medium 30%</th>
<th>Large 50%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Web Sites 10%</td>
<td>Microsoft</td>
<td>Oracle</td>
<td>IBM</td>
</tr>
</tbody>
</table>

- Not a big enough market to reach stated $100 million goal.

Figure Adapted from “Teaching Note for MySQL Open Source Database,” 6/1/04, Stanford GSB.
**My SQL: Growth Strategy**

- Many of these customers already using MySQL with websites
- Less emphasis on global organization
- Leverage SAP alliance
- Up against Microsoft.

<table>
<thead>
<tr>
<th></th>
<th>Small 20%</th>
<th>Medium 30%</th>
<th>Large 50%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enterprise</td>
<td>Microsoft</td>
<td>Oracle</td>
<td>IBM</td>
</tr>
<tr>
<td>wide data</td>
<td>Cost</td>
<td>Reliability</td>
<td>Support</td>
</tr>
<tr>
<td>90%</td>
<td></td>
<td>Longevity</td>
<td></td>
</tr>
<tr>
<td>Web Sites</td>
<td>My SQL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10%</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure Adapted from “Teaching Note for MySQL Open Source Database,” 6/1/04, Stanford GSB.
### My SQL: Growth Strategy

<table>
<thead>
<tr>
<th>Enterprise wide data 90%</th>
<th>Small 20%</th>
<th>Medium 30%</th>
<th>Large 50%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Web Sites 10%</td>
<td>Microsoft</td>
<td></td>
<td>Oracle IBM</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Reliability</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Scalability</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Support</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Longevity</td>
</tr>
</tbody>
</table>

- **Cost**
- **Maybe?**

- + builds on existing brand and strengths
- - Market not so big

Figure Adapted from “Teaching Note for MySQL Open Source Database,” 6/1/04, Stanford GSB.
Algorithms and protocols

Adapted from
David G. Messerschmitt
Algorithm

Specified sequence of steps that
- accomplish a designated task
- in a finite number of steps

Representation:
- simple algorithm: flowchart
- complicated algorithm: program
Example: one turn at monopoly

Start turn → Throw dice → Move token number of squares indicated on dice → Land on “go to jail”? → Yes → Move to “jail” square → Finish turn → No → Do not move; follow policies for square (like “pay rent”) → Finish turn
Algorithm building blocks

**Sequence**
- Start
  - Action
  - Action
  - Action
  - Finish

**Selection**
- Start
  - Decision
    - Action
    - Action
  - Finish

**Loop**
- Start
  - Test
    - Action
  - Finish

Programming languages support these three building blocks.
Protocol

- Distributed algorithm ...
- Realized by two or more modules to coordinate their actions or accomplish some shared task
- Module interoperability requires a protocol
  - Prescribed order of method invocations
  - Part of interface documentation
Monopoly players protocol

This is a protocol interaction diagram

Player 1

Player 2

One-turn algorithm

Time
Application and infrastructure

The application defines its own application-level protocols.

Internally, the network uses protocols to implement the services it provides.
Example:

HEADQUARTERS

Airline Dataserver

Airline Intranet

HHC Server

HHC

Wireless Link
Layered Protocols Example

HHC Server

HHC Server Application

Windows OS

Break Messages into Packets Networking Infrastructure

Send Packet

Acknowledge Packet

Link Level Protocol

Networking Infrastructure

HHC Application

Palm OS

Send Packet

Send Pass. Data As Message

Request Pass. Data

Application Level Protocol
Three simple protocols

One-way message: send-receive
Two-way interaction: request-response
Push: publish-subscribe
Send - Receive
Request - Response

Client

Server

request

response
Send - Acknowledge
**Example: HTTP** (Hyper Text Transfer Protocol)

1. User activates URL

2. HTTP request

3. HTTP response (embedded document)

4. Browser displays document (if HTML) or invokes “helper application”

HTTP server

HTTP client (browser)

HTML documents
Locating things

by

David G. Messerschmitt
Three ways of locating things

Name
- “Joe Bloe”

Address
- “1299 Hearst St, Berkeley, CA”

Reference
- “Postmaster of Berkeley CA”
Name

- Symbolic (character string) representation
- Easy for people to remember or guess
- Identifies, but
  - Does not locate directly
    - Distinction important for mobile entities
- Not unique: entities can have more than one name (called aliases)
Hierarchical names

Hierarchy makes names easier to remember or guess

Host domain names:

- “info.sims.berkeley.edu”
- designates administrative hierarchy

File names:

- “c:\My Documents\Docs\Resume.doc”
- designates folder hierarchy
Address

- Route or path to entity
  - is directly specified, or
  - can be inferred

- Independent of who or what is locating entity

- Topological specification
Example

Path from \textcolor{teal}{\text{teal}} \text{ to } \textcolor{brown}{\text{brown}} \text{ is } (R,D,D,D,R,R,R,R,R)

Is \textcolor{teal}{\text{teal}} \text{ (R,D,D,D,R,R,R,R,R)} \text{ an address?}
No! -- not an address, because it depends on starting point
Example

Address of (6,5) is (6,5)

Route from can be inferred
Reference

Abstract representation of an entity

Interaction is with representation
- infrastructure arranges redirection to actual entity
- especially appropriate for things that move

Example
- A Cell phone number is a *reference*.
- A Wired phone number is an *address*.
Name services

1. name
2. address or reference
3. interaction

Name service